The Dissertation Committee for Olivier Dubois-Matra
certifies that this is the approved version of the following dissertation:

Development of Multisensor Fusion Techniques with Gating Networks Applied to Reentry Vehicles

Committee:

Robert H. Bishop, Supervisor

Maruthi R. Akella

Wallace T. Fowler

Joydeep Ghosh

David G. Hull

Development of Multisensor Fusion Techniques with Gating Networks Applied to Reentry Vehicles

by

Olivier Dubois-Matra, Dipl.E., M.S.

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

August 2003

To my grandfather Joaquim.

Acknowledgments

I first would like to express my gratitude to my supervisor, Dr. Robert H. Bishop for his help and guidance on this work, for which he was able to secure funding from the Charles Stark Draper Laboratory, Inc. at MIT and the Jet Propulsion Laboratory at Caltech. His enthusiasm was a constant source of inspiration. I also would like to thank my committee members Dr. Maruthi R. Akella, Dr. Wallace T. Fowler, Dr. Joydeep Ghosh and Dr David G. Hull for reviewing my work, and for providing useful feedback. In addition, the precision Mars entry navigation project would not have been possible without the support of Dr. Todd Ely from JPL.

Research does not grow out of nothing, we always build on the work of our predecessors. I would like to mention Dr. Nassib Nabaa, for the tracking and estimation software he developed, and Dr. Wassim Samir Chaer and Dr. Timothy P. Crain for their pioneering work on applying gating networks to interplanetary navigation.

All these years I benefited from the support of several great friends, on both sides of the Atlantic, many of whom shared with me the ups and downs of graduate school. Great thanks thus to Michel, David, Calina, Mark and many others. And of course, all my thoughts to my family, for their love, support, and a great deal of patience.

Development of Multisensor Fusion Techniques with Gating Networks Applied to Reentry Vehicles

Publication No. _____

Olivier Dubois-Matra, Ph.D.
The University of Texas at Austin, 2003

Supervisor: Robert H. Bishop

The problem of model inaccuracy for Extended Kalman Filters (EKF) is addressed in the case of vehicle atmospheric entry tracking and navigation with a filter bank architecture, also called mixture-of-experts, regulated by gating network, which is then tested in two different applications.

First, a wind-frame based flight model is developed, which allows for maneuvers, and inclusion of atmospheric and gravity models. This level of complexity allows in theory for better estimation accuracy when used in an EKF, but the filter performance is in part dependent on the accuracy of the vehicle and environment models. The problem is how to deal with imperfect models. The approach treated here, which as already been applied in other domains, is to create a population of filters, each representing a particular modeling of the vehicle and/or environment. The discriminating device between the expert filters is a gating network, which is a simplified single-layer neural network learning in real-time with the help of the statistical information from the filters. The gating network is used to compute a weighted sum

of the state estimate from each filter, which is therefore an optimal estimate. The gating network can also be used as an hypothesis tester, which is the case in the first example.

The system was applied to the tracking and identification at high altitude of reentering spiraling objects accompanied by decoys. The object is being tracked at high altitude by three ground radars providing a variety of measurements which are treated in parallel by two filters, one being an expert tuned for the real target and the other tuned for the decoy. Experiments show that the regulated bank can rapidly correctly identify the object as being the real target.

The second application is precision Mars entry navigation, where the on-board navigation system of a maneuvering Mars lander used a bank of expert EKF, each processing inertial acceleration as measurement, and each designed around a specific realization of the imperfectly known atmospheric density profile. The objective here is less to identify the best performing model than optimizing the overall state estimate by combining the estimate from every filter. The system also periodically restarts the filters with the current optimal estimate so as to keep all the filters competitive during all of the descent. The result is that this mixture-of-experts does not perform better than a dead-reckoning scheme unless one of the density model happens to be relatively close from the real density profile, but that it is more robust than dead-reckoning to loss of data, and can readily adapt additional sources of measurements.

Table of Contents

Acknowledgments v

Abstract vi

List of Tables xi

List of Figures xii

Chapter 1. Introduction 1

1.1 The Need for Multisensor Fusion Techniques with Gating Networks Applied to Entry Navigation and Tracking 1

1.2 Previous Work 2

1.3 Presentation of the Current Research 4

Chapter 2. Extended Kalman Filter Bank Regulated by Gating Network 7

2.1 Mixture-of-Expert Architecture Overview 7

2.2 The Continuous-Discrete Extended Kalman Filter 9

2.3 The Gating Network and the Weight Update Procedure 11

2.3.1 Description of the Gating Network 11

2.3.2 Derivation of the Weight Update Procedure 16

2.3.3 Hierarchical Mixture-of-Experts 19

Chapter 3. Flight and Measurement Models 21

3.1 Flight Model 21

3.1.1 The Wind Frame 22

3.1.2 The Equations of Motion 25

3.2 Gravity and Atmospheric Models 28

3.2.1 Gravity Models 28

3.2.2 Atmospheric Models 29
3.3 Measurement Models 30
3.3.1 Ground Radar Tracking Model (Chapter 4) 30
3.3.2 Inertial Measurement Unit Model (Chapter 5) 31

Chapter 4. Tracking and Identification of a Maneuvering Reentry Vehicle from the Ground 35
4.1 Introduction 35
4.2 Definition of the aerodynamic coefficient 36
4.3 Tracking Simulation 39
4.3.1 States and Parameters Estimation 40
4.3.2 Entry Scenario Parameters 41
4.3.3 Results 42
4.3.4 Conclusion 58
4.4 Torsion Analysis 59
4.4.1 Torsion Analysis 60
4.4.2 Conclusion 67

Chapter 5. Mars Entry Precision Navigation 70
5.1 Introduction 70
5.1.1 Rational for Mars Entry Precision Navigation 70
5.1.2 Previous Examples of Atmospheric Entry Navigation with Conic Shell Spacecraft 73
5.1.3 Chapter Plan 75
5.2 Simulation Description 75
5.2.1 Generation of Reference Trajectories with SORT 75
5.2.2 Vehicle 80
5.2.3 Atmospheric Density Model 80
5.2.4 Individual Filter Description 83
5.2.5 Filter Bank Description 85
5.2.5.1 Fusion of Filter Covariances 86
5.3 Simulation Results 88
5.3.1 First Filter Bank 89

5.3.2 Second Filter Bank 96
5.3.3 Test of Robustness with Temporary Loss of IMU Data . 102
5.4 Conclusions 103

Chapter 6. Conclusions **108**

Appendices **111**

Appendix A. Derivation of the State Propagation Partials **112**

Bibliography **117**

Vita **123**

List of Tables

3.1 Constants for gravity model. (*) The Earth was assumed spherical in Chapter 4. 29

4.1 Constant values for MaRV tracking scenario 41
4.2 True initial values 41
4.3 Measurement error values for MaRV tracking scenario 42
4.4 Initial estimation error covariance values 42
4.5 Process noises 42
4.6 Final state estimation errors 43

5.1 Characteristics of the 2009 lander 80
5.2 Atmospheric density profile constants (Reference trajectory from [1]) 81
5.3 Coefficients for each filter in the bank 81
5.4 Process noise spectral density values in the RSW frame 85
5.5 Initial estimation error covariance values 85

List of Figures

2.1 General form of a single-layer gating network. 8

2.2 Details of a gating network neuron 11

2.3 Representation of the decision hyperplanes created by a gating network in a 3-D measurement space (first case). Decision hyperplane and measurement sequence generated by the corresponding model: real parameter value (blue), 50% (red), 200% (magenta), 10% (green) 14

2.4 Gating network gain histories: real parameter value (blue), 50% (red), 200% (magenta), 10% (green) 15

2.5 Representation of the decision hyperplanes created by a gating network in a 3-D measurement space (second case). Decision hyperplane and measurement sequence generated by the corresponding model: real parameter value (blue), 50% (red), 200% (magenta), 90% (green) 17

2.6 Gating network gain histories: real parameter value (blue), 50% (red), 200% (magenta), 90% (green) 18

3.1 Main reference frames 24

4.1 Trajectory with $\beta_m = 4\times10^4\ Kg/m^2$, with the first radar (star), second (square) and third (diamond). Projections of the radars and the trajectory along each axis are in black. 44

4.2 True altitude and surrounding atmospheric density for $\beta_m = 4 \times 10^4\ Kg/m^2$ 45

4.3 Bank angle, and lift and drag accelerations for $\beta_m = 4\times10^4\ Kg/m^2$ 45

4.4 Target position estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 46

4.5 Decoy position estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 46

4.6 Target velocity estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 47

4.7 Decoy velocity estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 47

4.8 Target acceleration estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 48
4.9 Decoy acceleration estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 48
4.10 Target λ and $\dot{\lambda}$ estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 49
4.11 Decoy λ and $\dot{\lambda}$ estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 49
4.12 Target ϕ and $\dot{\phi}$ estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 50
4.13 Decoy ϕ and $\dot{\phi}$ estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs 50
4.14 Trajectory with $\beta_m = 4 \times 10^3\ Kg/m^2$. Projections of the radars and the trajectory along each axis are in black. 51
4.15 True altitude and surrounding atmospheric density for $\beta_m = 4 \times 10^3\ Kg/m^2$ 51
4.16 Bank angle, and lift and drag accelerations for $\beta_m = 4 \times 10^3\ Kg/m^2$ 52
4.17 Target position estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 52
4.18 Decoy position estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 53
4.19 Target velocity estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 54
4.20 Decoy velocity estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 54
4.21 Target acceleration estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 55
4.22 Decoy acceleration estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 55
4.23 Target λ and $\dot{\lambda}$ estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 56
4.24 Decoy λ and $\dot{\lambda}$ estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 56
4.25 Target ϕ and $\dot{\phi}$ estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 57

4.26 Decoy ϕ and $\dot{\phi}$ estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs 57
4.27 Gating network gain histories ($\beta_m = 4 \times 10^4\ Kg/m^2$): real parameter value (blue), 50% (red) 58
4.28 Gating network gain histories ($\beta_m = 4 \times 10^3\ Kg/m^2$): real parameter value (blue), 50% (red) 59
4.29 Torsion terms with $\beta_m = 4.10^4$: τ_b (blue), τ_d (red) and τ_e (black). 62
4.30 Trajectory with $\beta_m = 4.10^4$ and $\dot{\phi} = 0$. 62
4.31 Cosine of the angle between ω and $\mathbf{F}$. The torsion ($\times 100$) has been superposed in red. 63
4.32 Torsion terms with $\beta_m = 4 \times 10^4\ Kg/m^2$: τ_{b_1} (blue) and τ_{b_2} (red). 65
4.33 Cosine of the angle between the lift and gravity vectors. The torsion ($\times 100$) has been added in red. 66
4.34 Lift (blue) and gravity (black), both in g's. τ_{b_2} added in red. 67
4.35 Lift (blue) and gravity (black), both in g's, for $\beta_m = 4 \times 10^3\ Kg/m^2$. $\tau_{b_2}(\times 100)$ added in red. 68
4.36 Detail of trajectory with $\beta_m = 4 \times 10^3\ Kg/m^2$ 68

5.1 True MCI position 77
5.2 True MCI velocity 77
5.3 True MCI non-gravitational acceleration 78
5.4 True altitude, relative velocity and atmospheric density . . . 78
5.5 True aerodynamic coefficients 79
5.6 True maneuvering angles and wind-frame accelerations 79
5.7 Actual (dotted line) and filter atmospheric density profiles down to 36 Km altitude (figures indicate time elapsed in seconds). Filter 1 : blue , 2 : red, 3 : magenta, 4 : green, 5 : gray, 6 : light blue. 82
5.8 Actual (dotted line) and filter atmospheric density profiles down from 36 Km altitude down to parachute deployment. Filter 1 : blue , 2 : red, 3 : magenta, 4 : green, 5 : gray, 6 : light blue. . 82
5.9 Differences between actual and filter atmospheric density profiles down to 36 Km altitude. Filter 1 : blue , 2 : red, 3 : magenta, 4 : green, 5 : gray, 6 : light blue. 83

5.10 Differences between actual and filter atmospheric density profiles down from 36 Km altitude down to parachute deployment. Filter 1 : blue , 2 : red, 3 : magenta, 4 : green, 5 : gray, 6 : light blue. 84
5.11 First filter bank: gating network weights 91
5.12 First filter bank: optimal RSW position estimation, averaged over 30 Monte Carlo runs 91
5.13 First filter bank: optimal RSW velocity estimation 92
5.14 First filter bank: optimal RSW acceleration estimation 92
5.15 First filter bank: optimal wind frame acceleration estimation . 93
5.16 First filter bank: additional data 93
5.17 Position estimation error covariance evolution from beginning of simulation to 70sec (magnified 3 times). True trajectory and its projection on the surface of Mars is in blue. Seconds are labeled in white. 94
5.18 Position estimation error covariance evolution from 80sec to end of simulation (magnified 3 times). True trajectory and its projection on the surface of Mars is in blue. Seconds are labeled in white. The shade of green is proportional to the atmospheric density. 95
5.19 Second filter bank: actual and filter atmospheric density profiles down to 36 Km altitude (figures indicate time elapsed in seconds) 97
5.20 Second filter bank: actual and filter atmospheric density profiles down from 36 Km altitude down to parachute deployment . . 97
5.21 Second filter bank: differences between actual and filter atmospheric density profiles down to 36 Km altitude 98
5.22 Second filter bank: differences between actual and filter atmospheric density profiles down from 36 Km altitude down to parachute deployment 98
5.23 Second filter bank: gating network weights 99
5.24 Second filter bank: optimal RSW position estimation, averaged over 30 Monte Carlo runs 99
5.25 Second filter bank: optimal RSW velocity estimation 100
5.26 Second filter bank: optimal RSW acceleration estimation . . . 100
5.27 Second filter bank: optimal wind frame acceleration estimation 101
5.28 Second filter bank: additional data 101
5.29 Temporary loss of data: gating network weights 103

5.30 Temporary loss of data: optimal RSW position estimation, averaged over 30 Monte Carlo runs 104
5.31 Temporary loss of data: optimal RSW velocity estimation . . 104
5.32 Temporary loss of data: optimal RSW acceleration estimation 105
5.33 Temporary loss of data: optimal wind frame acceleration estimation 105
5.34 Temporary loss of data: additional data 106

Chapter 1

Introduction

1.1 The Need for Multisensor Fusion Techniques with Gating Networks Applied to Entry Navigation and Tracking

Model-based navigation systems for aerospace vehicles can reach high levels of accuracy, assuming both the vehicle and the environment models are themselves an accurate representation of their real world counterparts. Estimators like the Kalman Filter (KF), developed by Kalman and Bucy [2, 3], and its extension to nonlinear systems, the extended Kalman filter (EKF) can use these models to effectively incorporate measurements to produce precise estimates of the vehicle state such as position, velocity and attitude. A lack of knowledge, however, in the flight dynamics or environment implies a less realistic flight model which leads to poorer estimation accuracies [4, 5]. Unknown parameters can be added to the state vector, provided some assumptions are made on their dynamics, but the system might become marginally observable or even non-observable.

Atmospheric entry (or re-entry in the case of Earth atmosphere) is a type of navigation problem where it is difficult to accurately model the environment [6]. Vehicles typically plunge from outside the atmosphere into successive layers, each with different density and temperature profiles; they encounter different flight regimes from hypersonic to subsonic, from newto-

nian to continuous flow. Winds vary in altitude, direction and strength. The exact atmospheric profile is itself constantly changing in space and time with periods going from hours to months. Complex and detailed models exist for the Earth atmosphere, but are computationaly intensive, not differentiable and thus difficult to implement in a filter. The problem is compounded in the case of a spacecraft entering another planetary atmosphere, in which case models are based on previous in-situ and remote observations, and no local information at the time of arrival is available to initialize the many parameters of the model.

Another problem is the model of the vehicle itself. The flight characteristics of a vehicle can be analyzed in detail through simulation and testing but the resulting model might be too complex to be implemented in a real time filter unless it is greatly simplified. In military applications (for example the tracking of a warhead ejected from a ballistic missile), little might be known about the vehicle itself and one of the tasks of the tracking system could be to identify the unknown object.

A significant source of modeling uncertainty is the process noise of the Kalman filter, which is often tuned to a satisfying value through an *ad hoc* process, and might alternatively be determined through adaptive filtering. Indeed, most examples in the literature considering adaptive filtering deal with uncertainties in process (and also measurement) noises, rather than dynamics or measurement model (a typical example being Mehra [7]).

1.2 Previous Work

Methods of *adaptive filtering* have been developed to allow for the filter to be modified in order to provide a closer fit to the real data. An early

classification of these methods was written by Mehra [7]. The classes are Bayesian, maximum likelihood, correlation and covariance matching methods. Most of those methods, however, are subject to theoretical and numerical limitations which restrict their domain of validity. One of the approaches is to have several filters, each working as an expert based on a certain model of the vehicle, the environment and / or the statistics of the measurement or process noise, in such a way that these experts together cover the range of possible (or at least, expected) representations of the real world (we call such set of filters a *mixture-of-experts*, or ME). What is then needed is a system that would give a measure of the likelihood that each filter represents the correct, or at the very least, the best model. This is the case for the Magill adaptive filter, a Bayesian scheme ([8, 9]). It does however suffer from numerical instability, and it makes the assumption that the real expert is among the population of experts being investigated.

Another approach is an offspring of neural network theory, and more precisely of the concept of modular networks. It is the *mixture-of-expert regulated by a gating network* [10–14]. The statistical weights are computed by a *gating network*, a single layer of simplified perceptrons, or *neurons*, with one neuron assigned per expert. Unlike traditional neural networks, no training is required before the utilization of the gating network, since it learns online by using statistical information from the filters themselves. The gating network approach compares favorably with the Magill coefficients approach [15–17].

Following these considerations, a strategy for developing a robust tracking (or navigation, if on-board) system would be to develop a series or bank of EKFs, each corresponding to a certain realization of the flight model (for example, different possible atmospheric profiles) and to have an on-line expert

system able to regulate in real time the bank by attributing statistical weights to each of the filters. Such a system has already be designed and tested for interplanetary navigation [15, 17, 18], and is able to identify model changes (such as measurement noise, solar radiation pressure and other unmodeled accelerations). It can be used both in real time and off-line, and can also, after selecting the best performing filter, use parameter optimization techniques (RQP, genetic algorithm) to tune the parameters of the selected filter.

1.3 Presentation of the Current Research

The goal of this dissertation is the development of a generic multisensor fusion system for entry (or re-entry) navigation which will use a hierarchical ME regulated by gating networks. A concise description of the underlying mathematics will be given in Chapter 2.

A generalized flight model is developed in Chapter 3 which represents a wide range of lift-generating bodies, provided there is no thrust applied. Employing the time-derivative of the acceleration, this model provides insight into the acceleration components, and can be used to separate the contributions to the in-plane and out-of-plane motion. This allows for the propagated trajectory to fit the true trajectory more precisely, assuming all parameters are known with precision, and it also gives access to variables such as the aerodynamic roll angle which are used for guidance. The generalized flight model will be tested in two different applications.

The first model, in Chapter 4, is the ground radar tracking of a re-entering maneuvering vehicle , where, in addition to estimating the position and the velocity of the target, the ME has to be able to identify the target through its aerodynamic characteristics from among a number of known tar-

gets. A current application, which will be the object of the example presented here, is the problem of high altitude identification and tracking of maneuvering re-entering warheads released alongside decoys of different masses and sizes [19]. The goal here is to show that the ME with the generalized flight model can perform these two functions : first, by identifying the target as either genuine or decoy,(as early as possible and as high as possible in the atmosphere) while the differentiating aerodynamic effects on warheads and decoys are extremely small. Second, by effectively tracking the target lower in the atmosphere, while it is engaged in a spiraling motion which is intended to make tracking and intercept more difficult.

The second model (Chapter 5) is an on-board, real-time navigation system for a Mars lander in the pre-parachute deployment phase, to be used in conjunction with an active guidance system which will need an accurate estimate of the state (position and related derivatives), the goal being to keep the final value of the state within certain bounds at parachute deployment. Due to the nature of the hypersonic and supersonic phases of the flight, the types of measurement available will be limited : the presence of the heat shield prevents the use of radar or laser ranging, the atmospheric ionization limits the use of radio navigation with any future ground or orbital beacon, and cost, weight and accuracy considerations could prevent the use of atmospheric sensors. In the mixture-of-experts, the EKFs incorporate inertial acceleration measurements to update the state vector. This is a departure from current dead-reckoning methods where inertial acceleration measurement are only propagated, along with estimated errors, which is a simpler, more robust method, but where navigation uncertainties can only grow with time. EKFs on the other hand incorporate measurements to reduce or at least limit state estimation errors,

at the cost of added complexity. This complexity is in part due to the necessity of using a flight and a measurement models in the filter equations, models which are based on assumptions made on the flight dynamics and environment conditions. In turn, an expert system such as the gating network can learn from the way filters with different models perform by observing how closely these models represent the current actual conditions. Each filter can be specialized in one aspect of the Martian atmospheric profile (typical density profile of some layer, high altitude wind), although in this work the focus will be on density profiles. The expert system will continuously attribute weights to the filters, and combine their estimates to provide an optimal state estimation. Preliminary results have been presented by Bishop, Dubois-Matra and Ely [20].

In these two chapters we will describe the organization of the hierarchical mixture of experts in both situations, with the different models for each experts, and the results of experiments performed to simulate the behavior of the HME over several Monte Carlo runs, with a discussion of the subsequent results. Chapter 6 will present the overall conclusions.

Chapter 2

Extended Kalman Filter Bank Regulated by Gating Network

A general description of the filter bank (or mixture-of-experts) structure and elements is presented here. We begin with an overview of the EKF bank and its components, followed by a review of the Extended Kalman Filter in the continuous propagation - discrete update form [21]. This form of the EKF is used because in the two scenarii studied measurements are acquired at discrete time interval (here, 0.1 *sec.*) while the environment varies continuously. The chapter concludes with a presentation of the regulating gating network derivation and a derivation of its online update procedure.

2.1 Mixture-of-Expert Architecture Overview

The filter bank contains L elements, each an EKF with its own parametrization, represented by a vector $\boldsymbol{\alpha}_i$ as illustrated in Figure 2.1. Dimension of the parameter vector for each filter can be different. The filters and the gating networks all receive the measurement vector $\boldsymbol{z}_k$ at the same time. In addition the gating network is provided with the residuals and residual covariances from each filter. Those are needed to compute the weights g_i, which are then used to compute the weighted sum of the state estimates. The weights are modified online in such a way as to maximize over time the probability weight for the best performing filter.

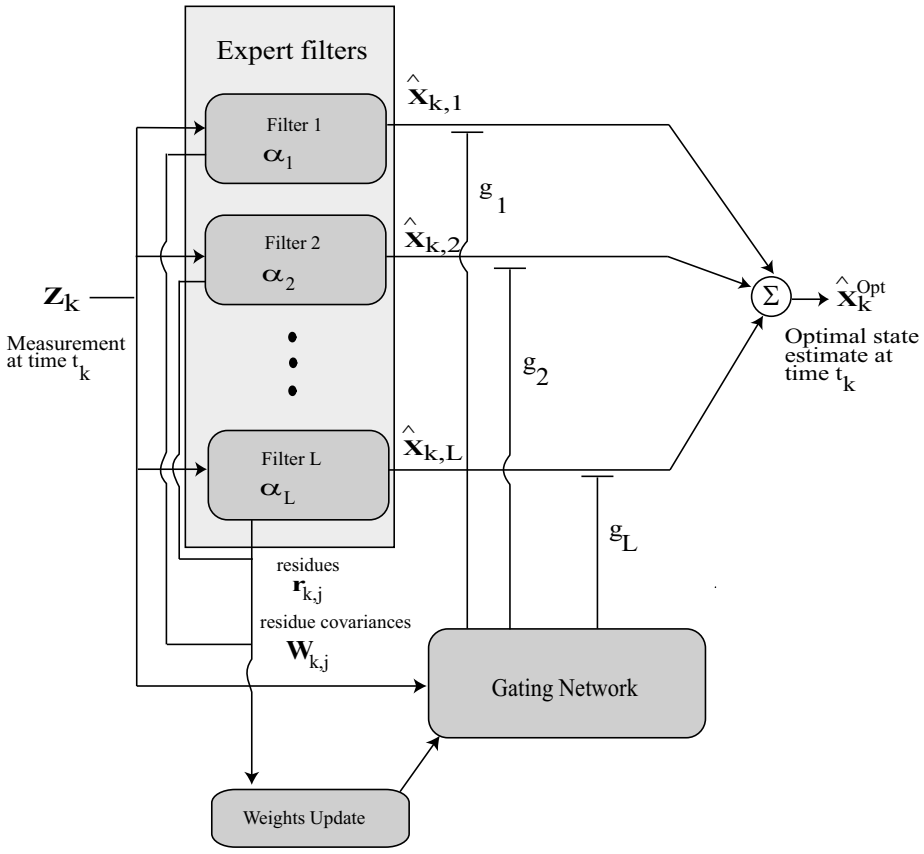

Figure 2.1: General form of a single-layer gating network.

2.2 The Continuous-Discrete Extended Kalman Filter

The system and measurement models used in the tracking algorithm include a white noise process $\boldsymbol{w}$, with zero mean and spectral density $\boldsymbol{Q}(t)$, and a measurement noise $\boldsymbol{\nu}_k$, which is a white random sequence of zero mean gaussian random variables with associated covariance matrix $\boldsymbol{R}_k$. We consider $\boldsymbol{R}_k$ as a characteristic of the measurement sources, while $\boldsymbol{Q}(t)$ is a tuning parameter for the filter. It is assumed that the process and measurement noises are statistically independent. The system and measurement models are

$$\begin{aligned} \dot{\boldsymbol{x}}(t) &= \boldsymbol{f}(\boldsymbol{x}(t), t) + \boldsymbol{w}(t) \\ \boldsymbol{z}_k &= \boldsymbol{h}_k(\boldsymbol{x}(t_k)) + \boldsymbol{\nu}_k. \end{aligned} \tag{2.1}$$

The vector $\boldsymbol{x}(t)$ represents the true state and $\boldsymbol{x}(t_k)$ is the state at t_k. The partials of both system and measurement vectors, $\boldsymbol{f}$ and $\boldsymbol{h}$ respectively, are needed for the propagation and update of the state vector and the error covariance matrix in the EKF formulation. Symbolically these are

$$\begin{aligned} \boldsymbol{F}(\hat{\boldsymbol{x}}(t), t) &= \left.\frac{\partial \boldsymbol{f}(\boldsymbol{x}(t), t)}{\partial \boldsymbol{x}(t)}\right|_{\boldsymbol{x}(t)=\hat{\boldsymbol{x}}(t)} \\ \boldsymbol{H}_k(\hat{\boldsymbol{x}}_k^-) &= \left.\frac{\partial \boldsymbol{h}_k(\boldsymbol{x}(t_k))}{\partial \boldsymbol{x}(t_k)}\right|_{\boldsymbol{x}(t_k)=\hat{\boldsymbol{x}}_k^-} \end{aligned} \tag{2.2}$$

where the superscript $\hat{}$ represents an estimate, and $-$ represents a pre-update value. The state estimate and estimation error covariance are propagated with

$$\begin{aligned} \dot{\hat{\boldsymbol{x}}}(t) &= \boldsymbol{f}(\hat{\boldsymbol{x}}(t), t) \\ \dot{\boldsymbol{P}}(t) &= \boldsymbol{F}(\hat{\boldsymbol{x}}(t), t)\boldsymbol{P}(t) + \boldsymbol{P}(t)\boldsymbol{F}^T(\hat{\boldsymbol{x}}(t), t) + \boldsymbol{Q}(t) \end{aligned} \tag{2.3}$$

for $t_{k-1} < t \leq t_k$, where t_k is the time of the next measurement. The EKF update occurs when the measurement is available at t_k via

$$\begin{aligned} \boldsymbol{K}_k &= \boldsymbol{P}_k^- \boldsymbol{H}_k^T(\hat{\boldsymbol{x}}_k^-)[\boldsymbol{H}_k(\hat{\boldsymbol{x}}_k^-)\boldsymbol{P}_k^- \boldsymbol{H}_k^T(\hat{\boldsymbol{x}}_k^-) + \boldsymbol{R}_k]^{-1} \\ \hat{\boldsymbol{x}}_k^+ &= \hat{\boldsymbol{x}}_k^- + \boldsymbol{K}_k[\boldsymbol{z}_k - \boldsymbol{h}_k(\hat{\boldsymbol{x}}_k^-)] \\ \boldsymbol{P}_k^+ &= [\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k(\hat{\boldsymbol{x}}_k^-)]\boldsymbol{P}_k^-. \end{aligned} \tag{2.4}$$

where $\boldsymbol{K}_k$ is the Kalman gain. The algorithm starts with an initial state estimate and state error covariance. In addition, define the measurement residual $\boldsymbol{r}_k$ and the residual covariance $\boldsymbol{W}_k$ as

$$\begin{aligned} \boldsymbol{r}_k &= \boldsymbol{z}_k - \boldsymbol{H}_k \hat{\boldsymbol{x}}_k \\ \boldsymbol{W}_k &= \boldsymbol{H}_k \boldsymbol{P}_k^- \boldsymbol{H}_k^T + \boldsymbol{R}_k, \end{aligned} \tag{2.5}$$

respectively. The residual and residual covariance are employed in the gating networks (discussed in the next section) to determine the gating weights. As stated previously, the process noise matrix $\boldsymbol{Q}(t)$ is a tuning factor for the filter; its value is determined mostly *ad hoc* through simulation. We choose it as a diagonal matrix whose only non-zero elements are the highest derivative of any given state. The purpose of the process noise is to take into account unmodeled dynamics and error sources in order to prevent the filter from rejecting measurements after a while. Each expert in the filter bank will employ the EKF model given in Eq. 2.3 and 2.4. The filter parametrization occurs in two primary (and related) components : first the values of $\boldsymbol{R}_k$ and $\boldsymbol{Q}(t)$ can be varied from filter to filter , and second, the dynamics model and the measurement models, represented by $\boldsymbol{f}(\boldsymbol{x}(t), t)$ and $\boldsymbol{h}_k(\boldsymbol{x}(t_k))$, respectively, can have different model parameters, which implies that the partials $\boldsymbol{F}(\hat{\boldsymbol{x}}(t), t)$ and $\boldsymbol{H}_k(\hat{\boldsymbol{x}}_k^-)$ vary by filter.

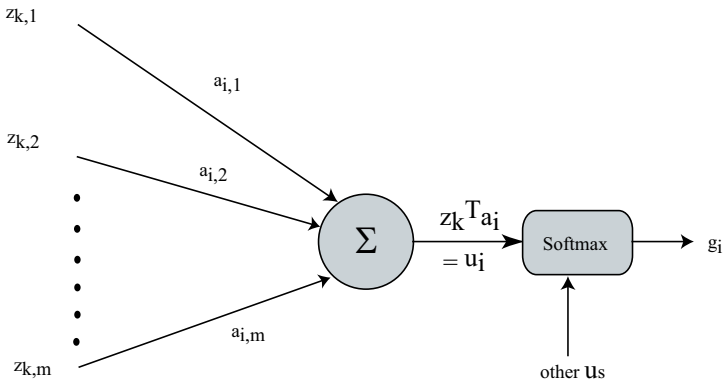

Figure 2.2: Details of a gating network neuron

2.3 The Gating Network and the Weight Update Procedure

2.3.1 Description of the Gating Network

The gating network is basically a single layer of *cells*, or *neurons*, which are elementary pattern classifiers similar to the perceptron described in Haykin [22], and illustrated in Figure 2.2. The perceptron receives a vector of inputs (in the present case, measurements) through its *synapses* and it computes a weighted sum of the input, which is then passed through a hard limiter (normally a step function), where it is compared to a threshold, thus giving a binary output to the perceptron. The goal is to divide the input space by a hyperplane, with one half of the space containing the inputs matching the required pattern [23]. For this, the neuron weights have to be updated to achieve a correct orientation of the hyperplane through a training procedure such as backpropagation, where a series of inputs associated with known outputs are presented to the perceptron, and the results are used to update the weights accordingly. A layer of these neurons will create a set of hyperplanes which will partition the input space into overlapping regions.

The neurons used in the gating network differ from the perceptrons in two ways. First, instead of a step function, a *softmax* ([24, 25]), or sigmoid-like function, is applied to the output signal, with a threshold set to zero. The use of a differentiable function instead of a step function will make possible the following derivations. It does not change the steady-state response of the neuron as a classifier [26]. Second, the input weights of the neurons will be updated on-line in accordance with statistical information continuously provided by the filters, in contrast to the usual heuristic methods of supervised training where the networks are presented with training sets before being used with real inputs.

The gating weight, or output of the i^{th} neuron, is

$$g_i = \frac{e^{u_i}}{\sum_{j=1}^{K} e^{u_i}}, \tag{2.6}$$

which is the softmax function, where

$$u_i = \boldsymbol{z}_k^T \boldsymbol{a}_i, \tag{2.7}$$

with $\boldsymbol{z}_k$ being the measurement vector at time t_k, and $\boldsymbol{a}_i$ is the input weight vector. The scalar u_i is called the *activation*, and is a measure of the probability for the i^{th} expert to be the correct match for this measurement. The application of the softmax function has three goals: a differentiable function for the derivation of the update equations, as stated before; an emphasis on the best performing filter to the detriment of all others, since the differences between the gating weights are now exponential; finally a normalization,

$$\sum_{i=1}^{L} g_i = 1, \tag{2.8}$$

which makes it possible to treat the gs as probabilities.

As said before, the purpose of a neuron is to separate the measurement in space in two with a hyperplane, defined by the equation

$$\boldsymbol{z}_k^T \boldsymbol{a}_i \quad = \quad 0, \tag{2.9}$$

and thus orthogonal to $\boldsymbol{a}_i$. Measurements occurring in the half-space toward which $\boldsymbol{a}_i$ is pointing are validated by the neuron, the others are rejected. Initially, all the weight vectors have arbitrary values. The goal of the weight update procedure is to maximize the gating weight of the best performing filer by updating the weight vector in such a way as to maximize the activation for this filter, and to do so not just by increasing the norm of $\boldsymbol{a}_i$, but also by aligning it with the measurement vector. Ideally, after some amount of time, the hyperplanes are oriented in such a way that only the hyperplane corresponding to the best performing filter validates the measurements. A graphical example of this adaptation will be shown in Chapter 4.

To illustrate this, an example with a three dimension measurement space is presented. This is similar to the case described in Chapter 4, with a target being tracked with range, elevation and azimuth measurements. To better represent the behavior of the gating network, the filter bank has four experts (instead of two like in Chapter 4), each tuned for a target with a specific mass. The measurement tracks are shown in blue. The measurement tracks that would be generated by an object with a different mass are also shown in different colors for comparison (in order, red, magenta and green). The decision hyperplanes and the $\boldsymbol{a}_i$ vectors for each filter are represented with the corresponding colors. In the first case shown in Figure 2.3, the first filter (blue) has the true value for the mass, while the others have a mass twice (red), half (magenta) and a tenth (green) of the real mass. The configuration

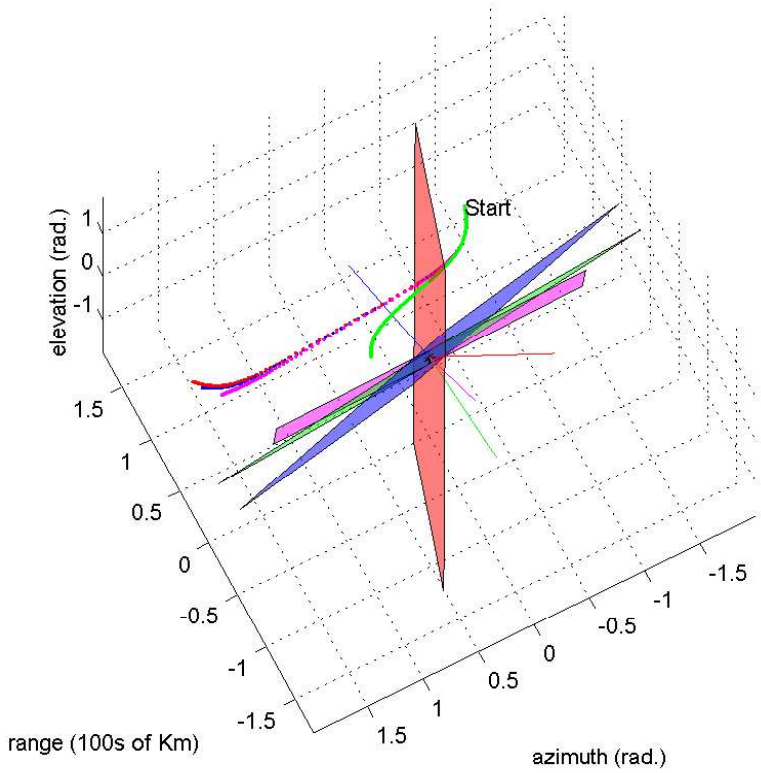

Figure 2.3: Representation of the decision hyperplanes created by a gating network in a 3-D measurement space (first case). Decision hyperplane and measurement sequence generated by the corresponding model: real parameter value (blue), 50% (red), 200% (magenta), 10% (green)

of the hyperplanes shown here is the one obtained at the end of the simulation (all input weights across the gating network were initialized with the same value at the beginning). It can be seen that only $\boldsymbol{a}_1$ is pointing toward the measurements, and that therefore only the neuron for the first filter will validate the measurements received. This can be seen in the gating weights performances shown in Figure 2.4 where g_1 approaches unity.

In the second case, however, the fourth filter works with a mass hy-

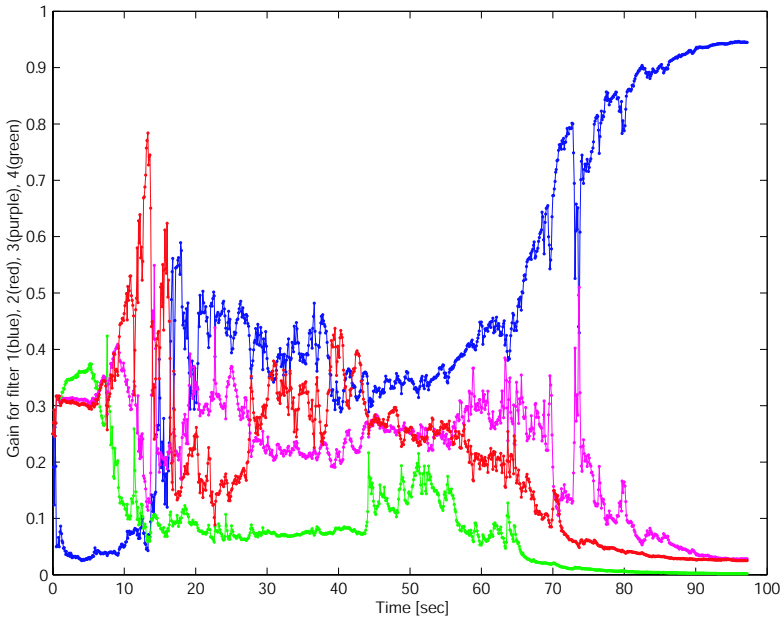

Figure 2.4: Gating network gain histories: real parameter value (blue), 50% (red), 200% (magenta), 10% (green)

pothesis much closer to the actual (90 percent of the real mass value), as seen in Fig. 2.5. It can be hypothesized that the gating network will have a tougher time deciding between filters 1 and 4. It does seem to be the case when looking at the measurement space. The decision half-spaces for these filters overlap. However, the first filter still outperforms the other three filters, as seen in Fig. 2.6, although not as clearly as in the previous case. So, even if the decision half-planes are overlapping, the differences between the activations, amplified by the softmax function, can be enough for the gating network to rank the filters correctly.

2.3.2 Derivation of the Weight Update Procedure

The weights g_i can be considered as *a priori* probabilities for the corresponding filters to represent the model generating the next measurement. Thus, the probability prediction for $\boldsymbol{z}_k$ is

$$f(\boldsymbol{z}_k) \;=\; \sum_{j=1}^{L} f(\boldsymbol{z}_k \mid \alpha_j) g_j \tag{2.10}$$

where the conditional probability for the i^{th} filter, being dependant on the statistics of its residues, is Gaussian, with mean $\boldsymbol{r}_k$ and covariance $\boldsymbol{W}_k$:

$$f(\boldsymbol{z}_k \mid \alpha_i) \;=\; \frac{1}{\sqrt{2\pi \mid \boldsymbol{W}_{i,k} \mid}} \exp \frac{1}{2} \boldsymbol{r}_{i,k}^T \boldsymbol{W}_{i,k}^{-1} r_{i,k}. \tag{2.11}$$

In order to make the derivation easier, the logarithm of the probability $f(\boldsymbol{z}_k)$ is used

$$l \;=\; \ln f(\boldsymbol{z}_k). \tag{2.12}$$

The *a posteriori* probability for the i^{th} filter is

$$h_i \;=\; \frac{f(\boldsymbol{z}_k \mid \alpha_i) g_i}{\sum_{j=1}^{L} f(\boldsymbol{z}_k \mid \alpha_j) g_j}. \tag{2.13}$$

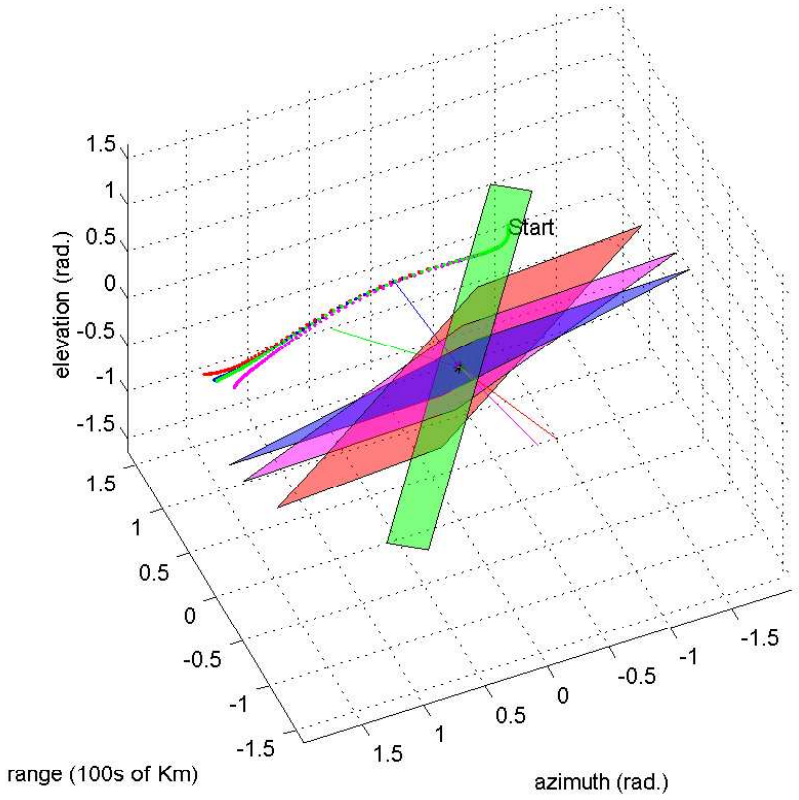

Figure 2.5: Representation of the decision hyperplanes created by a gating network in a 3-D measurement space (second case). Decision hyperplane and measurement sequence generated by the corresponding model: real parameter value (blue), 50% (red), 200% (magenta), 90% (green)

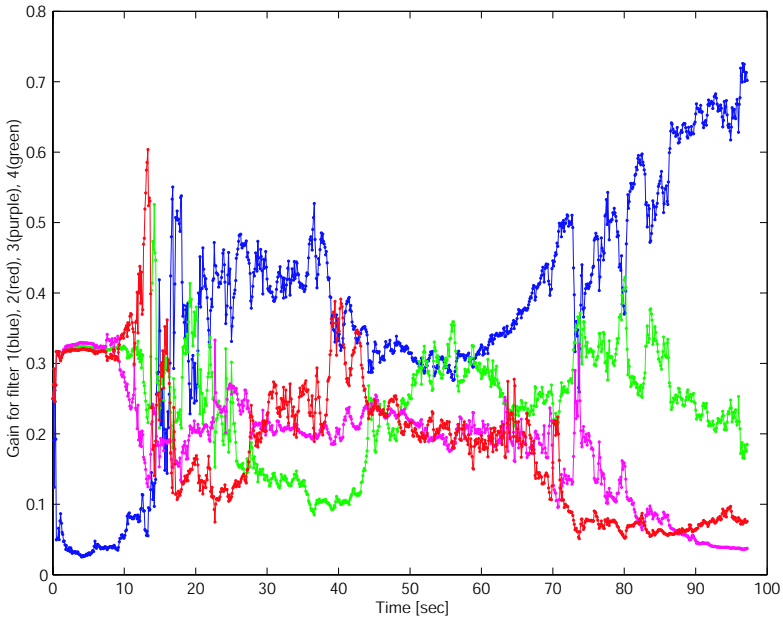

Figure 2.6: Gating network gain histories: real parameter value (blue), 50% (red), 200% (magenta), 90% (green)

The goal being to maximize l with respect to the a_i weights, the following partials are computed:

$$\frac{\partial l}{\partial \boldsymbol{a}_i} = \frac{\partial l}{\partial u_i}\frac{\partial u_i}{\partial \boldsymbol{a}_i} = (h_i - g_i)\boldsymbol{z}_k, \tag{2.14}$$

which gives the gradient direction in which to update each $\boldsymbol{a}_i$, the exact modifier for those being determined with an arbitrary (and heuristically determined) learning rate η (typically less than unity). The update step is given by

$$\Delta \boldsymbol{a}_i = \eta\frac{\partial l}{\partial \boldsymbol{a}_i} = \eta(h_i - g_i)\boldsymbol{z}_k, \tag{2.15}$$

thereby yielding

$$\boldsymbol{a}_i \leftarrow \boldsymbol{a}_i + \eta(h_i - g_i)\boldsymbol{z}_k. \tag{2.16}$$

2.3.3 Hierarchical Mixture-of-Experts

The mixture-of-experts scheme can be extended in such a way that each expert is itself a regulated mixture-of-experts. This hierarchical mixture-of-experts architecture (or HME) was first proposed by Jordan and Jacobs [13]. Each base-level bank can be specialized in finding the optimal value of a specific set of parameters, and the upper-level gating network regulates between the specialized banks. This architecture works well for hypothesis testing, where, for example, each base-level bank represents a specific event affecting the model, and each expert inside a bank represents specific parameter values for that event. This approach was applied to the interplanetary navigation problem with the goal of detecting unmodeled accelerations [18], where the type of acceleration and the corresponding parameters were treated at different level of the HME. Each example in this dissertation require only one level

of bank, since the filters represent numerical variations over the same set of parameter. Taking into account qualitatively different models (such as having both variation in the atmospheric density and the wind profile in the Mars navigation example) might have to be achieved with a multi-layer HME. The goal here is to demonstrate the feasibility of applying ME to atmospheric entry problems, and this added level of complexity shall be reserved for future studies. Therefore, we will limit ourselves here to a single-level mixture-of-experts approach.

Chapter 3

Flight and Measurement Models

Modeling is a crucial step in navigation applications. A balance has to be reached between accuracy and complexity on one hand, and simplicity (and therefore, computational speed) on the other hand. Computational speed is important in the case of an on-board, real-time navigation system. Also, it is desirable to have differentiable model equations, which can be implemented in a Kalman filter without requiring numerical differentiation - an important source of numerical errors.

These requirements should not prevent the equations from reproducing the dynamics of the real states and measurements up to a satisfying precision. The ultimate test for the model is to compare it to real data, or if that is not possible, to compare it with more complex (but less implementable) models. It must be recalled that the ultimate goal is to have the mixture-of-experts selects the best matching filter among a pool of imperfect filters.

3.1 Flight Model

The following flight model is intended to accurately represent an unpowered vehicle with lifting capacity and with aerodynamic bank angle ϕ and total angle of attack α as control. It can be implemented either as a *6-state* filter (position and velocity coordinates only) or a *9-state* filter (including ac-

celeration). The use of time derivatives beyond the acceleration offers some deeper insight into the dynamics of atmospheric flight, as shown in Bishop [27], and Mehrotra [28]. The advantages and disavantages of the 9-state filter will be discussed later. The wind frame is presented first.

3.1.1 The Wind Frame

The basis of the flight model is a center-of-gravity centered wind-frame depicted in Figure 3.1. The lift and drag vector will lie along main directions in this frame, thus allowing for simpler dynamic equations and easy definition of the control angles. From now on, $\boldsymbol{r}_I$, $\boldsymbol{v}_I$ and $\boldsymbol{a}_I$ denote respectively the position, velocity and non-gravitational acceleration of the vehicle in a planet-centered inertial frame.

The rotation vector of the planet in the inertial frame is

$$\boldsymbol{\Omega} = [0 \quad 0 \quad \Omega]^T \tag{3.1}$$

with Ω as the rotation rate. The *relative velocity* vector resulting from the motion of the vehicle through the atmosphere revolving at the same angular speed as the rest of the planet (assuming here there is no wind involved) is

$$\boldsymbol{v}_r = \boldsymbol{v} - \boldsymbol{\Omega} * \boldsymbol{r}, \tag{3.2}$$

where $*$ represents a cross-product. The wind-frame vectors $\boldsymbol{e}_i^w$ are defined as follows: the first vector is pointing in the direction of $\boldsymbol{v}_r$, that is

$$\boldsymbol{e}_1^w = \frac{\boldsymbol{v}_r}{V_r} \tag{3.3}$$

where V_r is the norm of $\boldsymbol{v}_r$. The second vector is defined with respect to a certain arbitrary reference vector $\boldsymbol{h}$. The convention for $\boldsymbol{h}$ may vary according

to the application. Therefore, we write

$$\boldsymbol{e}_2^w = -\frac{\frac{\boldsymbol{v}_r}{V_r} * \boldsymbol{h}}{\left\| \frac{\boldsymbol{v}_r}{V_r} * \boldsymbol{h} \right\|}. \tag{3.4}$$

For ground tracking of re-entry vehicles, $\boldsymbol{h}$ can be selected to be the north pole axis $\boldsymbol{U}_z$[6]. Otherwise, it is logical to choose $\boldsymbol{h}$ as the radius vector $\boldsymbol{r}_I$; in this case, $\boldsymbol{e}_2^w$ is orthogonal to the relative velocity direction. Singularities in both cases are unrealistic, since they would imply descent from a perfect polar orbit ($\left\| \frac{\boldsymbol{v}_r}{V_r} * \boldsymbol{U}_z \right\| = 0$) or a vertical fall ($\left\| \frac{\boldsymbol{v}_r}{V_r} * \boldsymbol{r}_I \right\|$=0). In the following discussion, we will assume, without loss of generality, that $\boldsymbol{h}$ is equal to $\boldsymbol{r}_I$. The third vector completes the frame and is pointing in the overall "up" direction:

$$\boldsymbol{e}_3^w = \boldsymbol{e}_1^w * \boldsymbol{e}_2^w. \tag{3.5}$$

If no thrusting is taking place, the only non-gravitational accelerations are the lift and drag. The drag vector is parallel to $\boldsymbol{e}_1^w$ and pointing in the opposite direction. The lift vector is contained in the $(\boldsymbol{e}_2^w, \boldsymbol{e}_3^w)$ plane. Define ϕ as the angle between the lift vector $\boldsymbol{L}$ and $\boldsymbol{e}_3^w$, positive in the clockwise direction when looking in the direction of $\boldsymbol{e}_1^w$. We refer to ϕ as the aerodynamic bank angle, or simply the bank angle. The same angle is not to be confused with the body roll angle. Although related, those two angles are distinct.

The bank angle is a primary guidance variable. It indicates the rotation of the lift vector around the relative velocity vector. The other guidance variable is the angle of attack. It does not appear explicitly in the equations of motion above, but it is implicitly contained in the aerodynamic coefficients. Since the two applications studied here employ different models for the aerodynamic coefficients, they will be described in their respective chapters.

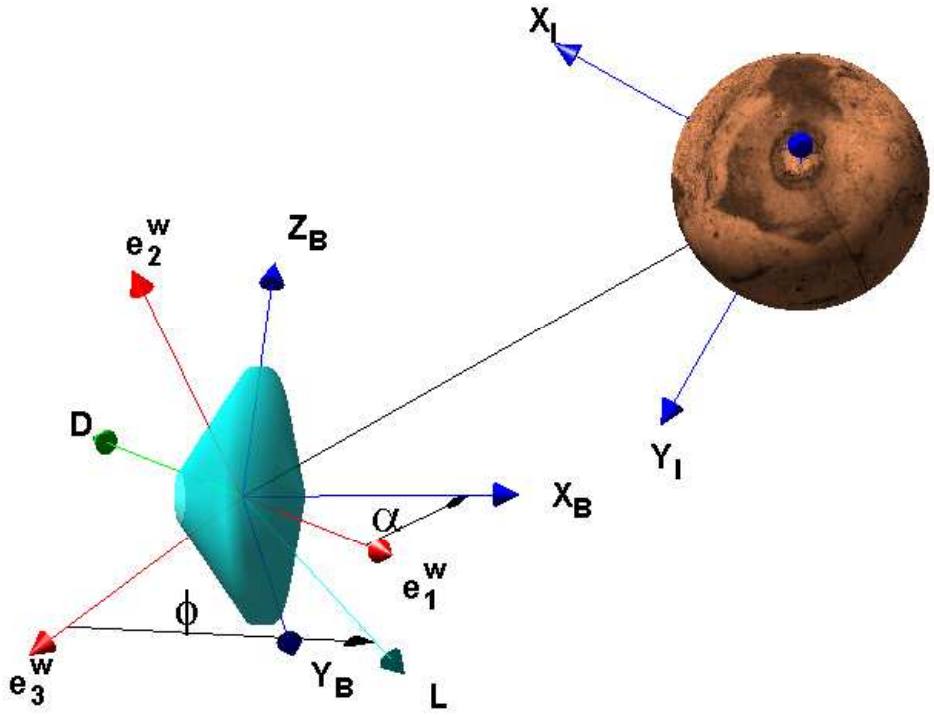

Figure 3.1: Main reference frames

3.1.2 The Equations of Motion

The equation of motions in the planet centered inertial frame can be written employing the previously defined wind frame. Two approaches have been investigated with regard to the length of the state vector. When only position and velocity are considered (6 states), the resulting equations are

$$\dot{\boldsymbol{r}} = \boldsymbol{v} \tag{3.6}$$

$$\dot{\boldsymbol{v}} = \boldsymbol{a} + \boldsymbol{g}, \tag{3.7}$$

with

$$\boldsymbol{a} = -D\boldsymbol{e}_1^w + L\left(-\boldsymbol{e}_2^w \sin\varphi + \boldsymbol{e}_3^w \cos\varphi\right), \tag{3.8}$$

L and D being the lift and drag acceleration, respectively, and $\boldsymbol{g}(\boldsymbol{r})$ is the gravity acceleration (whose exact model need not be specified as of now).

Alternatively, non-gravitational acceleration can be used as a state (9 states), changing the equations to

$$\dot{\boldsymbol{r}} = \boldsymbol{v} \tag{3.9}$$

$$\dot{\boldsymbol{v}} = \boldsymbol{a} + \boldsymbol{g} \tag{3.10}$$

$$\dot{\boldsymbol{a}} = \overbrace{\left[\boldsymbol{\omega}^w * \boldsymbol{a}\right]}^{(a)} + \overbrace{\left[\dot{\varphi}\ \boldsymbol{e}_1^w * \boldsymbol{a}\right]}^{(b)} - \overbrace{\left[\dot{D}\boldsymbol{e}_1^w\right]}^{(c)} + \underbrace{\dot{L}\left[-\boldsymbol{e}_2^w \sin\varphi + \boldsymbol{e}_3^w \cos\varphi\right]}_{(d)}, \tag{3.11}$$

and

$$\boldsymbol{\omega}^w = \frac{\boldsymbol{v}_r \times \dot{\boldsymbol{v}}_r}{\|\boldsymbol{v}_r\|^2}. \tag{3.12}$$

and $\boldsymbol{v}_r$ is the relative velocity. Again, the potential occurrence of singularity is limited to the unlikely case where the relative airspeed is zero.

Both the 6 and 9 states models have their advantages and drawbacks. Including the aerodynamic acceleration as a state leads to a more complex

propagation model and partials. Since the time derivatives for ϕ, D and L are involved, a better knowledge of the dynamics of those quantities is required (which also means knowledge of the time derivatives of the aerodynamic coefficients like C_L and C_D). Assuming C_L and C_D are constant or piecewise constant might lead to important estimation errors when the actual coefficient changes rapidly, which does happen when the atmospheric flow regime goes from free molecule flow to continuum [6]. Also more states means more possibilities for integration errors.

On the other hand, more information is available through the extra states, which can be useful for filtering, prediction, and guidance. Some of that information appears in the third derivative equation in Eq. 3.11, where each term on the right-hand side corresponds to a single cause of the *jerk*, or change in the acceleration, of the vehicle:

- (a) is the contribution of the motion inside a plane perpendicular to $\boldsymbol{\omega}^w$. The norm of $\boldsymbol{\omega}^w$ is the curvature of the trajectory inside that plane, which is a fixed plane under certain conditions, such as no banking, constant lift and drag, and uniform gravity field. This set of hypothesis is required for the *coordinated turn model* described in Bishop & Antoulas ([27]).

- (b) is the contribution to the out-of-plane motion by the rotation of the lift vector, whose angular rate is $\dot{\varphi}$. A constant, non-zero value of $\dot{\varphi}$ leads to a spiraling motion as described in Chapter 4.

- (c) is the change caused by variation of the drag. Since the drag vector is contained in the curvature plane, this term does not contribute to out-of-plane motion.

- (d) is the change caused by variation of the lift, which can lead to out-of-plane motion.

Another advantage of the 9-states model is that using certain measurement types can lead to simpler measurement partials when implemented in a Kalman filter. For example, with inertial acceleration measurements, which are measured in a body-fixed frame, the measurement equation is

$$\boldsymbol{y} = \boldsymbol{C}_I^B \boldsymbol{a}_I, \tag{3.13}$$

with $\boldsymbol{C}_I^B$ being the transformation matrix from inertial to body frame. The resulting measurement partial equations will be much simpler and less error-prone with 9 states. In particular, for the 9-state filter we have

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{X}} = \begin{bmatrix} \mathbf{0}_{3\times3} & \mathbf{0}_{3\times3} & \boldsymbol{C}_I^B \end{bmatrix}. \tag{3.14}$$

and for the 6 state filter

$$\frac{\partial \boldsymbol{y}}{\partial \boldsymbol{X}} = \left[C_I^B \frac{\partial \boldsymbol{a}_I}{\partial \boldsymbol{X}} \right]. \tag{3.15}$$

Alternatively, it is also possible to select the total acceleration, instead of the aerodynamic acceleration only, as a state. In this case, the knowledge of the time derivative of the gravity acceleration $\boldsymbol{g}$ is necessary. That is, Eq. 3.9-3.11 become

$$\dot{\boldsymbol{r}} = \boldsymbol{v} \tag{3.16}$$

$$\dot{\boldsymbol{v}} = \boldsymbol{a}_T \tag{3.17}$$

$$\dot{\boldsymbol{a}}_T = [\boldsymbol{\omega}^w * (\boldsymbol{a}_T - \boldsymbol{g})] + [\dot{\varphi}\ \boldsymbol{e}_1^w * (\boldsymbol{a}_T - \boldsymbol{g})] - \left[\dot{D}\boldsymbol{e}_1^w\right] + \dot{L}\left[-\boldsymbol{e}_2^w \sin\varphi + \boldsymbol{e}_3^w \cos\varphi\right] + \dot{\boldsymbol{g}} \tag{3.18}$$

Although this representation was used in Chapter 4, its only clear advantage over the "aerodynamic acceleration only" approach seems to be to explicitly represent the out-of-plane motion caused by a non-uniform gravity field. Both implementations were tested and found to be applicable.

3.2 Gravity and Atmospheric Models

This will be only a brief discussion of the gravity and atmospheric models. They were left in a general form in the previous equation on purpose, since they depend on the flight application.

3.2.1 Gravity Models

Typically, it is possible to increase the accuracy of a gravity model through the addition of spherical harmonics ([29]). A model limited to the oblateness J_2 was found to be sufficiently accurate for filtering purposes, and will be used for both applications in Chapters 4 and 5. The gravity including J_2 is given by

$$\boldsymbol{g} = -\frac{GM}{R^3}\begin{bmatrix} r_x(1-\frac{3}{2}J_2(\frac{a_e}{R})^2(5\frac{r_z^2}{R^2}-1)) \\ r_y(1-\frac{3}{2}J_2(\frac{a_e}{R})^2(5\frac{r_z^2}{R^2}-1)) \\ r_z(1+\frac{3}{2}J_2(\frac{a_e}{R})^2(-5\frac{r_z^2}{R^2}+3)) \end{bmatrix}, \tag{3.19}$$

where $\boldsymbol{r} = [r_x, r_y, r_z]$ is the planet centered inertial position, with norm R, and the constants are as presented in Table 3.1.

Constants	$GM(m^3/s^2)$	$a_e(m)$	J_2
Earth	3.986005×10^{14}	6.37814×10^6	$0(*)$
Mars	4.28221×10^{13}	3.396×10^6	1.0826×10^{-3}

Table 3.1: Constants for gravity model. (*) The Earth was assumed spherical in Chapter 4.

3.2.2 Atmospheric Models

The general expressions for the lift and drag accelerations and their time derivatives are

$$\begin{aligned} L &= C_L q_\infty \frac{S}{m} \\ D &= C_D q_\infty \frac{S}{m} \\ \dot{L} &= \frac{S}{m}\left(\dot{C}_L q_\infty + C_L \dot{q}_\infty\right) \\ \dot{D} &= \frac{S}{m}\left(\dot{C}_D q_\infty + C_D \dot{q}_\infty\right), \end{aligned} \tag{3.20}$$

where m is the mass of the vehicle, S is the reference area, or maximum cross-section area [6], and with the definition of the dynamic pressure as

$$q_\infty = \frac{1}{2}\rho V_r^2. \tag{3.21}$$

The parameters C_L and C_D are the lift and drag coefficients, respectively. These coefficients cannot normally be assumed to be constants, since the vehicle flight usually goes through both hypersonic and supersonic flight regimes. The two examples to be studied later each use a different model for the aerodynamic coefficients. In Chapter 4 the concept of *induced drag* will be introduced, while in Chapter 5 C_D and C_L will in fact be assumed to be constants to simplify the filter implementation, although more realistic simulation should make these coefficients dependent on both total angle of attack and Mach number.

Different atmospheric density profiles will also be used in each chapter. In Chapter 4 both the model for the real trajectory generation and the filter model are a simple exponential model. More advanced models could be employed in future simulations, but it is legitimate in any case to assume that the filter atmospheric model is accurate, since in this type of ground-based application *a priori* detailed knowledge of the current atmospheric density profile is available and can be provided to the filter. This is not the case for the Mars entry navigation problem in Chapter 5, which is why the filter models do not try to reproduce the entire exact true atmospheric profile, but instead numerical variations of the same model (a two-layer exponential model) are used to create a population of density models large enough to approximate the real density profile.

3.3 Measurement Models

3.3.1 Ground Radar Tracking Model (Chapter 4)

The measurements are taken from 3 radars site somewhere on the surface of the Earth (we arbitrarily chose a location in the Marshall Islands, the same used in Cardillo ([30]). One of the radar is the center of the local East-North-Up ($\boldsymbol{I}, \boldsymbol{J}, \boldsymbol{K}$) frame in which the measurements are defined. The others are 20 degrees east, 10 degrees north and 10 degrees east, 40 degrees north of the first radar, respectively. The radars take measurements at a frequency of 10 Hz. Each measurement provides range, range rate, azimuth and elevation. If $\boldsymbol{Pn}$ is the coordinate vector of the radar in Earth-centered frame, then the

measurements are defined by:

$$\begin{aligned} \boldsymbol{s} &= \boldsymbol{r} - \boldsymbol{Pn} \\ \rho &= \| \boldsymbol{s} \| \\ \dot{\rho} &= \frac{\dot{\boldsymbol{r}} \odot \boldsymbol{s}}{\| \boldsymbol{s} \|} \\ azi &= \arccos \frac{(\boldsymbol{s}*\boldsymbol{K}).\boldsymbol{I}}{\|\boldsymbol{s}*\boldsymbol{K}\|} \\ elev &= \pi/2 - \arccos \frac{\boldsymbol{s}.\boldsymbol{K}}{\|\boldsymbol{s}*\boldsymbol{K}\|} \end{aligned} \tag{3.22}$$

which gives us the following measurement vector at measurement time t_k:

$$\boldsymbol{z_k} = [\rho_k, \dot{\rho}_k, azi_k, elev_k]. \tag{3.23}$$

It is assumed that the position of the radars is known accurately, that the measurements are processed centrally, and that there is no measurement registration problem (i.e. the three separate radar returns at any given time will be identified as coming from the same object at the same time).

In the simulation, the measurements are corrupted by additive white noise. The values for the standard deviations of the noise are listed in 4.3.

3.3.2 Inertial Measurement Unit Model (Chapter 5)

The main and possibly only source of measurement for an onboard atmospheric entry navigation system is the inertial measurement unit, or IMU, a cluster of gyroscopes and accelerometers which provide data on the vehicle attitude and non-gravitational accelerations. The IMU, either of a platform or strapdown type, is located within the vehicle body reference frame and provide measurement within its own case frame. Both the angular rate of the body with respect to the inertial frame and the acceleration measurements

provided by the IMU are susceptible to be corrupted by a variety of noises, misalignments and other errors. In the dead-reackoning method of navigation, the locally measured accelerations are transformed into inertial acceleration and then integrated into position and velocity. Since this is an open loop process, there is no correction for errors in either attitude or acceleration data. The filter presented here, however, can some way compensate for these errors since the acceleration measured in body frame is a filter measurement corrupted by noise whose statistics are known, and the inertial acceleration is a filter state to be estimated.

The actual computation algorithm from IMU measurements to attitude, position and velocity information is complex, as presented, for example, by Savage([31, 32]) in the case of strapdown IMUs. Several simplifications have been made in the simulation, as listed here :

- The body and case frames coincide, and are both located at the center of gravity of the vehicle. Realistically, the IMU unit is normally located somewhere else for engineering reasons, and its frame is not necessarily aligned with the body frame.

- We assume perfect knowledge of the angular rate provided by the gyroscopes. It is integrated to provide the quaternion representing the orientation of the body frame in the inertial frame[6]. The quaternion carries the necessary information to form the transformation matrix from case frame to inertial frame $\boldsymbol{T}_C^I$. With the quaternion defined as

$$\dot{\boldsymbol{Q}} \quad = [\boldsymbol{q} \quad q_4]^T \quad = [q_1 \quad q_2 \quad q_3 \quad q_4]^T , \tag{3.24}$$

with

$$\boldsymbol{q} \in \mathbb{R}^3, q_4 \in \mathbb{R},$$

the transformation matrix is

$$\boldsymbol{T}_C^I = \boldsymbol{I}_{3\times 3} + 2q_4\boldsymbol{S}(\boldsymbol{q}) + 2\boldsymbol{S}^2(\boldsymbol{q}), \tag{3.25}$$

where $\boldsymbol{S}$ is the skew symmetric matrix operator, defined as

$$\boldsymbol{a} * \boldsymbol{b} = \boldsymbol{S}(\boldsymbol{a})\boldsymbol{b} \quad \forall \quad \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^3.$$

If $\boldsymbol{\omega}$ represents the angular rate, the quaternion propagation equation is

$$\dot{\boldsymbol{Q}} = \tfrac{1}{2}\boldsymbol{B}(\boldsymbol{\omega})\boldsymbol{Q}, \tag{3.26}$$

with

$$\boldsymbol{B}(\boldsymbol{\omega}) = \begin{bmatrix} 0 & \omega_z & -\omega_y & \omega_x \\ -\omega_z & 0 & \omega_x & \omega_y \\ \omega_y & -\omega_x & 0 & \omega_z \\ -\omega_x & -\omega_y & -\omega_z & 0 \end{bmatrix}. \tag{3.27}$$

The initial value of the quaternion, $\boldsymbol{Q}_0$, is known exactly. Realistically, the angular rate is corrupted by various sources, leading to the following representation for the measured angular rate:

$$\boldsymbol{\omega}_m = (\boldsymbol{I} + \boldsymbol{S}_g)(I + \boldsymbol{\Gamma}_g)(\boldsymbol{\omega} + \boldsymbol{b}_g + \epsilon_g), \tag{3.28}$$

where $\boldsymbol{b}_g$ is the gyro bias, $\boldsymbol{S}_g$ is the diagonal scale factor error matrix, and Γ_g is the zero-diagonal nonorthogonality error matrix, with all the coefficients being gaussian random constants. The noise ϵ_g is a gaussian random process. For this simulation, we assume that $\boldsymbol{\omega}_m = \boldsymbol{\omega}$.

- We assume that acceleration measurements are corrupted by gaussian noise only. Realistically, the acceleration measured in the case frame is corrupted by various sources, leading to the following representation:

$$\boldsymbol{a}_m = (\boldsymbol{I} + \boldsymbol{\Upsilon}_a)(I + \boldsymbol{\Xi}_a)(\boldsymbol{a} + \boldsymbol{b}_a + \epsilon_a), \tag{3.29}$$

where $\boldsymbol{b}_a$ is the accelerometer bias, Ξ_a is the diagonal scale factor error matrix, and Υ_a is the zero-diagonal nonorthogonality error matrix, with all the coefficients being gaussian random constants. The noise ϵ_a is a gaussian random process. For this simulation, we assume that $\boldsymbol{a}_m = \boldsymbol{a} + \epsilon_a$.

Chapter 4

Tracking and Identification of a Maneuvering Reentry Vehicle from the Ground

4.1 Introduction

In this chapter, the mixture-of-expert systems is used to track an unknown Maneuvering Reentry Vehicle (MaRV) from a network of ground-based radar stations, and additionally to identify it among a choice of several possible targets. An example of an application is the problem in ballistic missile defense of target identification, when an enemy missile releases several decoys in addition to the warhead. Decoys can have the same radar and/or thermal properties than the warhead, and therefore can fool sensors while still above the atmosphere, but different mass and aerodynamic properties can make them distinguishable during reentry. Thus, the ability of the mixture of experts to act as an hypothesis tester will be extremely important. This will be tested in a scenario where the gating network will have to decide between filters working with a different hypothesis for the ballistic coefficient of the MaRV.

Another aspect of the reentry problem that will be studied in this chapter is the possibility offered by the flight model to easily reproduce spiraling trajectories, which can be typical of the behavior of some types of ballistic missile evasion maneuvers, as described in Lewis and Postol [19]. More generally, the class of spiraling MaRV might include any type of vehicle using this type of behavior for stabilization purpose. A look into the cause of the

spiraling movement, and of the potential importance of the torsion of the trajectory, and its possible use for guidance purposes, is presented at the end of this chapter.

4.2 Definition of the aerodynamic coefficient

The detailed expressions of the aerodynamic coefficients were left unresolved in Chapter 3, since they are application-dependent. In this chapter, the "zero-lift drag coefficient" approach of Regan and Anandakrishnan [6] will be used.

Recall that lift and drag acceleration magnitude are defined as

$$\begin{aligned} L &= C_L q_\infty \frac{S}{m} \\ D &= C_D q_\infty \frac{S}{m}. \end{aligned} \tag{4.1}$$

Two quantities need to be defined now: the *zero-lift drag coefficient* C_{D_0} is the drag coefficient value when no lift is generated by the MaRV. The *critical lift coefficient* C_L^* is the lift coefficient at the point of maximum lift-to-drag ratio. Thus, $(\frac{C_L}{C_L^*})^2$ is the *induced drag*, and

$$C_D = C_{D_0}\left[1 + \left(\frac{C_L}{C_L^*}\right)^2\right]. \tag{4.2}$$

We see that the drag increases as the vehicle generates lift.

Alternatively, C_D can be described in the *drag polar form* as

$$C_D = C_{D_0} + KC_L^n, \tag{4.3}$$

where K is a constant, and n is vehicle-dependent. Dividing C_L by both sides

of Eq.

$$\frac{C_L}{C_D} = \frac{C_L}{C_{D_0} + KC_L^n}. \tag{4.4}$$

Computing the extremum with respect to C_L (and therefore, at $C_L = C_L^*$) yields

$$K = \frac{C_{D_0}}{(n-1)C_L^{*n}}. \tag{4.5}$$

The resulting lift-to-drag ratio is

$$\left(\frac{C_L}{C_D}\right)_{max} = \frac{(n-1)C_L^*}{nC_{D_0}}. \tag{4.6}$$

A value for n has to be specified. In the hypersonic region, Regan and Anandakrishnan [6], the range of values is given by

$$n = -0.09567\left(\frac{C_L}{C_D}\right)_{max} + 2.235, \tag{4.7}$$

but usually, for a vehicle with rotational symmetry, a value of $n = 2$ is chosen. In this case

$$C_{D_0} = \frac{C_L^*}{2\left(\frac{C_L}{C_D}\right)_{max}}. \tag{4.8}$$

Dividing again C_L by both sides and rearranging, we obtain

$$C_L = 2C_{D_0}\left(\frac{C_L}{C_D}\right)_{max}\left(\frac{C_L}{C_L^*}\right). \tag{4.9}$$

The lift-to-drag ratio is particularly important, since it is related to the angle of attack. It is important for guidance applications to be able to estimate the lift-to-drag ratio, and also for tracking where this type of information can

help to predict the immediate behavior of the target. Define the ratio of C_L to C_L^* as

$$\lambda = C_L/C_L^*. \tag{4.10}$$

Another important definition is that of *ballistic coefficient* given by

$$\beta_m = \frac{m}{C_{D_0}S}. \tag{4.11}$$

We can assume that $C_{D_0}S$ is constant. This is not necessarily the case when the flight profile covers wide airspeed range, but is a good approximation if only the hypersonic region is considered. The ballistic coefficient β_m is then a characteristic of the vehicle, and can therefore be used to identify it.

In term of the fundamental parameters C_{D_0}, λ and $\left(\frac{C_L}{C_D}\right)_{max}$, the final expressions for C_L and C_D are

$$\beta_m = \frac{m}{C_{D_0}S},$$

$$\begin{aligned} C_L &= 2C_{D_0}\left(\frac{C_L}{C_D}\right)_{max}\lambda \\ C_D &= C_{D_0}\left(1+\lambda^2\right), \end{aligned} \tag{4.12}$$

The corresponding time derivatives are

$$\begin{aligned} \dot{C}_L &= 2C_{D_0}\left(\frac{C_L}{C_D}\right)_{max}\dot{\lambda} \\ \dot{C}_D &= 2C_{D_0}\lambda\dot{\lambda}, \end{aligned} \tag{4.13}$$

which shows that the drag coefficient varies both when there is lift ($\lambda \neq 0$) or when there is a variation in the lift magnitude ($\dot{\lambda} \neq 0$).

4.3 Tracking Simulation

The purpose of this simulation is to illustrate the capacity of the gating network to identify a spiraling target at high altitude, early during the reentry phase, using measurements from three radar stations which provide information on the motion of the object through radiometric means. The radars do not have visual capacity, hence cannot directly identify the targets. For the sake of simplicity, and without loss of generality, only one object will be tracked at a time. In actuality, tens of decoys might be released alongside the target. It is assumed then that the radar stations have been able to separate the objects and are now focusing on identifying each of them. A sophisticated simulation was developed employing an extended Kalman filter, and a regulated bank of these EKFs. The goal of the bank is two-fold: first, to track the target with accuracy all along its trajectory. This implies determining, not just its current position and velocity, but also additional parameters, such as spiraling frequency. The second goal is to identify the target as a real target or as a decoy as soon as possible in the upper atmosphere.

The equations of motion developed for the MaRV and presented above were used to generate realistic trajectories with MATLAB[1], using a fourth order Runge-Kutta integration scheme. Error-free measurement sequences are also computed. The simulation, also written in MATLAB, generate sequentially several time the same tracking scenario with randomly initial state estimates and measurement noise sequences, so as to generate Monte Carlo runs. Within each Monte Carlo run, at every time step, and for each filter in the bank, the state estimate is first updated with the current measurement, and

[1]MATLAB is a registered trademark of TheM̃athWorks,Ĩnc.

then propagated along with the state error covariance. The propagation equations in this application are the same as the one used for the real trajectory generation (the uncertainty come from the lack of knowledge of the value of the parameters in the model). The integration is also performed with a fourth order Runge-Kutta integrator. Measurement and measurement residuals from each filter are used to compute the gating weights on-line. At the end of each runs, the state estimation results are averaged with those from the previous runs, the initial state estimates and the measurement noises are randomly reinitialized, and the next run begins.

4.3.1 States and Parameters Estimation

By "states" we define the position $\boldsymbol{p}$, velocity $\boldsymbol{v}$, and acceleration $\boldsymbol{a}$, of the MaRV, expressed in Earth-centered inertial coordinates. By "parameters" we define any other value being estimated, whether it is constant or not. States and parameters form the state vector to be estimated.

The flight model as presented in Chapter 3 has 3 degrees of freedom, plus two control parameters: λ, which controls the aerodynamic coefficients, and ϕ, which controls the orientation of the lift vector. The knowledge of these two values is required to estimate the motion of the MaRV. The parameters of interest therefore are λ, its time derivative $\dot{\lambda}$, and the orientation angle of the lift vector in the body frame ϕ and its time derivative $\dot{\phi}$. The state vector is thus

$$\boldsymbol{X} = [\boldsymbol{r}, \boldsymbol{v}, \boldsymbol{a}, \lambda, \dot{\lambda}, \phi, \dot{\phi}]. \tag{4.14}$$

C_{D_0}	$(C_L/C_D)_{max}$
0.033	1

Table 4.1: Constant values for MaRV tracking scenario

λ_0	$\dot{\lambda}_0$	ϕ_0	$\dot{\phi}_0$
2	0	0 rad	1 rad/sec

Table 4.2: True initial values

4.3.2 Entry Scenario Parameters

The initial position of the MaRV in the radar local frame was $s_0 = [-60, 60, 110]$ km. It is moving horizontally at an initial speed of 2000 m/sec. The object is tracked for about 100 seconds by the three radars as it plunges toward the ground. The simulation is stopped while the MaRV is above 50 Km, because interception typically need to occur above 40 Km [19], which would leave some margin to track the identified targets.

Actual parameter values for the physical properties of the target are difficult to obtain. Therefore, we invented two scenarios. In the first case, we assume $\beta_m = 4 \times 10^3$ Kg/m^2, and in the second case $\beta_m = 4 \times 10^4$ Kg/m^2. In each case, the decoys have half the value of β_m as the target. Other constants of importance are given in Table 4.1, and initial conditions are in Table 4.2.

The simulation generates noisy radar measurements at 10 Hz, with the measurement noise errors given in Table 4.3. We assume the measurement errors are known exactly and the environment and filter covariance are therefore the same. The error covariance for the initial guess estimate of the state are in Table 4.4. The process noise spectral density values used in the filters are given in Table 4.5.

Range Error	Range rate error	Angle Error
0.1 m	0.1 m/s	1 $mrad$

Table 4.3: Measurement error values for MaRV tracking scenario

$\boldsymbol{r}$	$\boldsymbol{v}$	$\boldsymbol{a}$	λ	$\dot{\lambda}$	ϕ	$\dot{\phi}$
400 m	50 m/s	5 m/s^2	.5	.02 s^{-1}	π rad	.3 rad/sec

Table 4.4: Initial estimation error covariance values

4.3.3 Results

The tracking results for the case with $\beta_m = 4\times10^4\ Kg/m^2$ are shown in Figures 4.1 -4.13. These results were averaged over 30 Monte Carlo runs, each with normally distributed random initial state estimation errors and measurement noise sequences. The same results for $\beta_m = 4 \times 10^3\ Kg/m^2$ are shown in Fig. 4.14-4.26. Altitude and atmospheric density (Fig. 4.2 and 4.15) and aerodynamic forces and bank (Fig. 4.3 and 4.16) are also represented. In each of the state estimation plots, the blue curve represents the state estimation error, and the red and green curves are the square root of the corresponding diagonal element of the state error covariance matrix, which is an indication of the *a priori* performance of the filter. Final state estimation error norms in position, velocity and acceleration are also displayed in Table 4.6.

Several remarks can be made on the estimation results. First, every state estimation sequence starts with a "learning" phase, where both estima-

${Q_{acc}}^2$	${Q_\lambda}^2$	${Q_{\dot\lambda}}^2$	${Q_\phi}^2$	${Q_{\dot\phi}}^2$
$10^{-4}\ m^2s^{-5}$	$10^{-5}\ s^{-1}$	$10^{-3}\ s^{-3}$	$10^{-5}\ rad^2s^{-1}$	$10^{-3}\ rad^2s^{-3}$

Table 4.5: Process noises

β_m	4.10^4 (target)	4.10^4(decoy)	4.10^3 (targ.)	4.10^3(dec.)
Position (m)	0.028	0.039	0.072	0.313
Velocity (m/s)	0.053	0.028	0.008	0.148
Acceler. (m/s^2)	0.046	0.033	0.046	0.115

Table 4.6: Final state estimation errors

tion and covariance errors increase rapidly, then slowly diminish (this is not the case for λ and $\dot{\lambda}$, in Fig. 4.11 and 4.24, as described later). Then after reaching an optimal phase during the middle of the simulation, the estimation performance slowly decreases. This is most visible in the position estimation plots in Fig. 4.4,4.5, 4.17 and 4.18. This is due to the fact that the MaRV get closer to the radars, and also is in the best position possible for efficient tracking (roughly inside the triangle shaped by the three radar) at mid-trajectory. Also, the oscillations due to the spiralling motion of the MaRV appears in the last seconds of the simulation.

The second observation is that there are apparently few differences between the state estimates of the two filters (they are more visible with the smaller value of β_m), and that these differences do not seem to clearly indicate that one of the filter is tracking the target better than the other. This is where the gating network is crucial. A learning rate of 0.01 was used. In both cases, by the 100 second mark, it identifies the object as being the real target with a likelihood of almost 80%, and the network starts leaning into that direction well before that point (Fig. 4.27 and 4.28). In the second case, with a lower β_m, aerodynamic forces have an earlier impact, and detection starts occurring before 60 seconds. There is, however, a slight loss in the weight assigned to the correct filter between 80 and 100 seconds, which has yet to be explained.

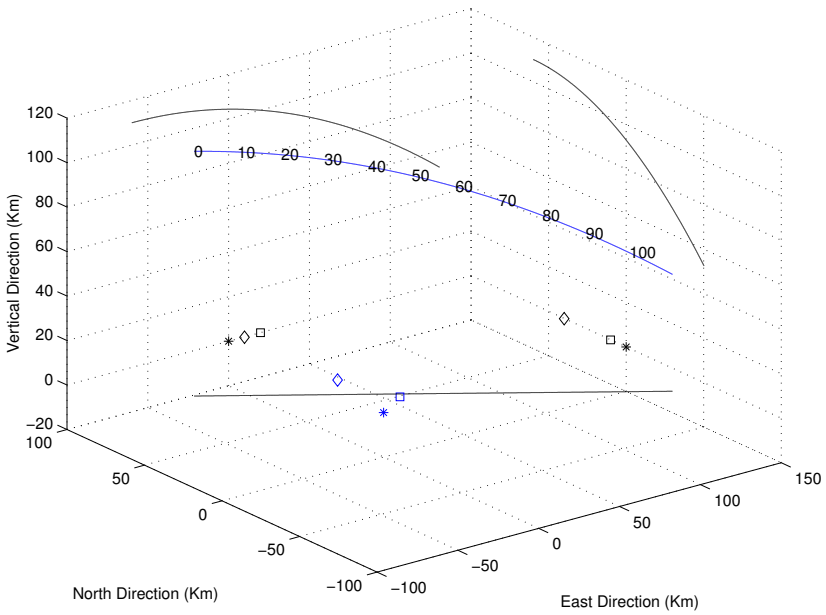

Figure 4.1: Trajectory with $\beta_m = 4 \times 10^4\ Kg/m^2$, with the first radar (star), second (square) and third (diamond). Projections of the radars and the trajectory along each axis are in black.

Finally, λ and $\dot{\lambda}$ appear to be marginally observable. The same can be said of the bank angle initially, although the bank rate estimate does seem to converge toward the end of the tracking period, when aerodynamic forces start to increase sharply. This is not surprising, since at the altitude during which the identification takes place the aerodynamic forces are still extremely small. The parameters λ and ϕ are important in the lower part of the trajectory, where a non-zero bank angle rate can produce a spiraling motion. The tracking of the target in the lower atmosphere should be the subject of future studies.

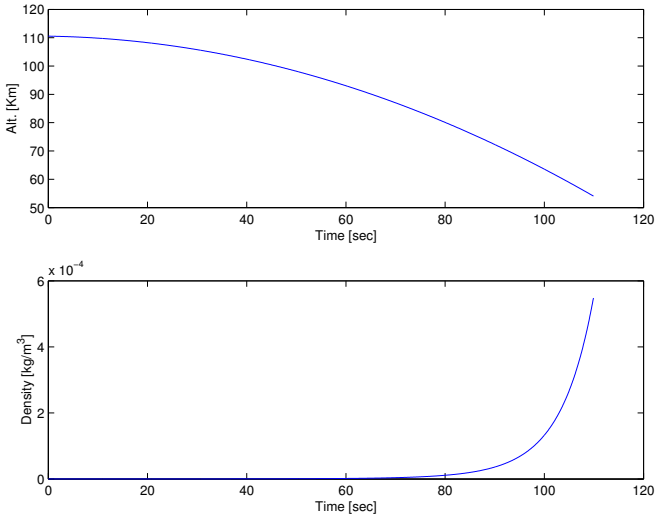

Figure 4.2: True altitude and surrounding atmospheric density for $\beta_m = 4 \times 10^4\ Kg/m^2$

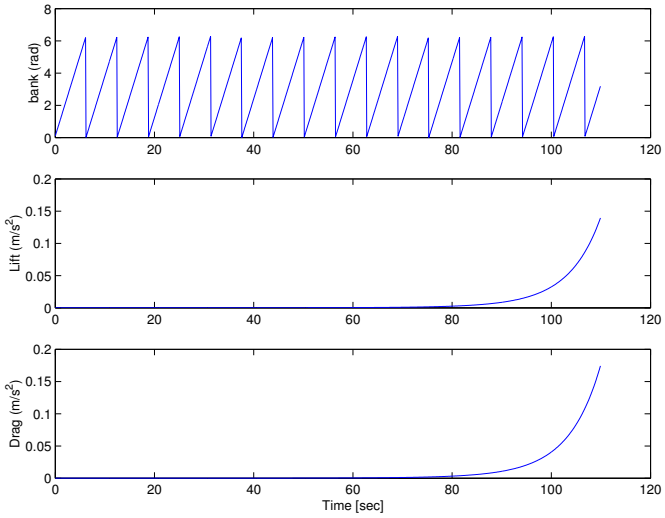

Figure 4.3: Bank angle, and lift and drag accelerations for $\beta_m = 4 \times 10^4\ Kg/m^2$

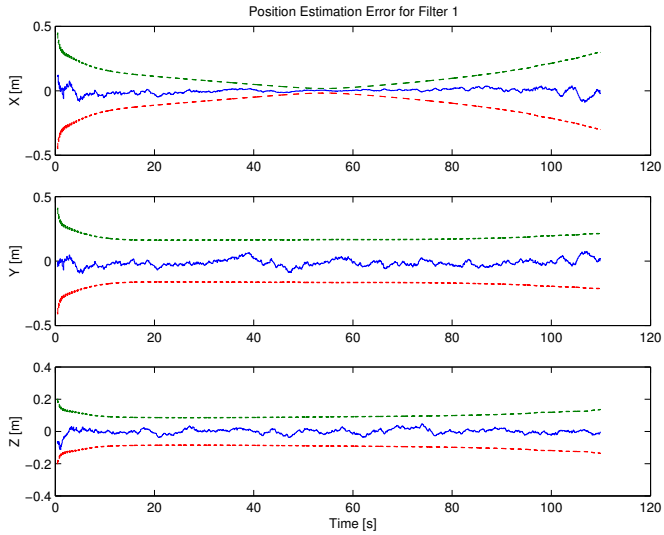

Figure 4.4: Target position estimation errors for $\beta_m = 4{\times}10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs

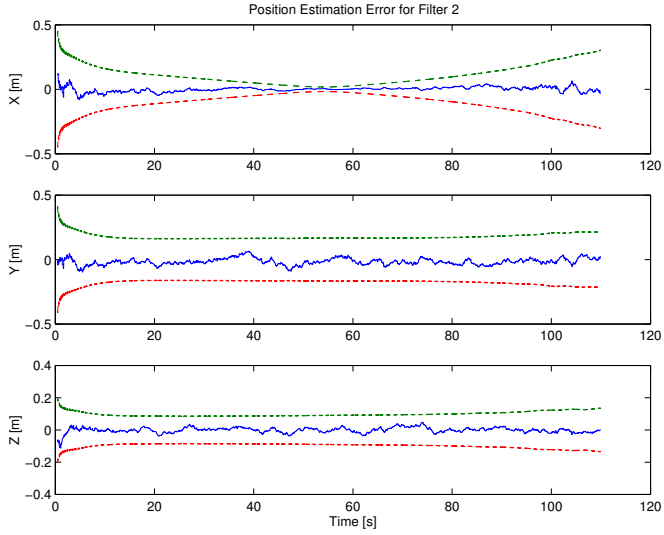

Figure 4.5: Decoy position estimation errors for $\beta_m = 4{\times}10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs

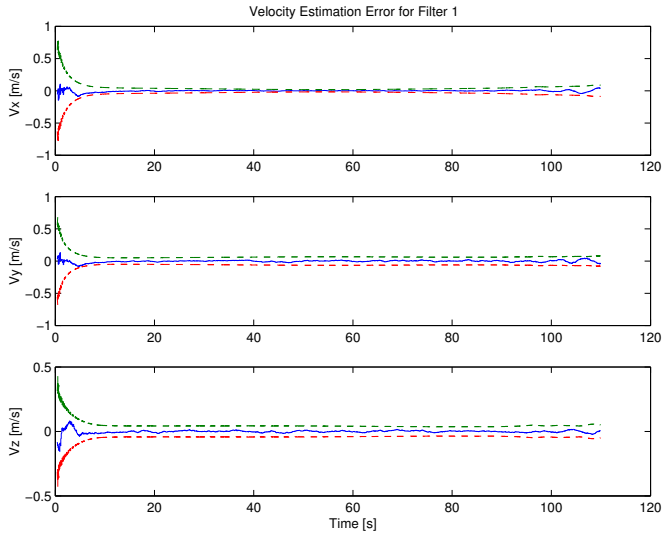

Figure 4.6: Target velocity estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs

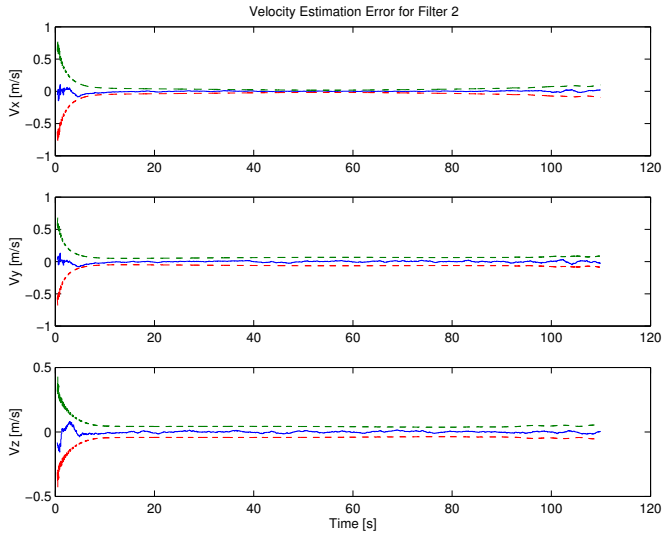

Figure 4.7: Decoy velocity estimation errors for $\beta_m = 4 \times 10^4\ Kg/m^2$, averaged over 30 Monte Carlo runs

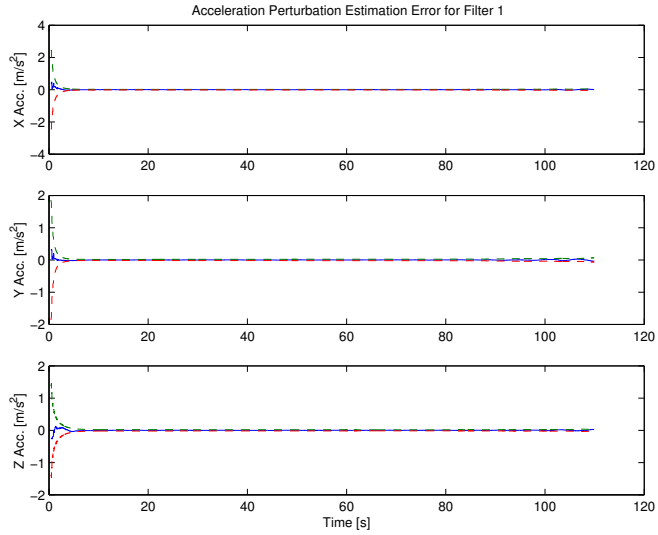

Figure 4.8: Target acceleration estimation errors for $\beta_m = 4 \times 10^4 \ Kg/m^2$, averaged over 30 Monte Carlo runs

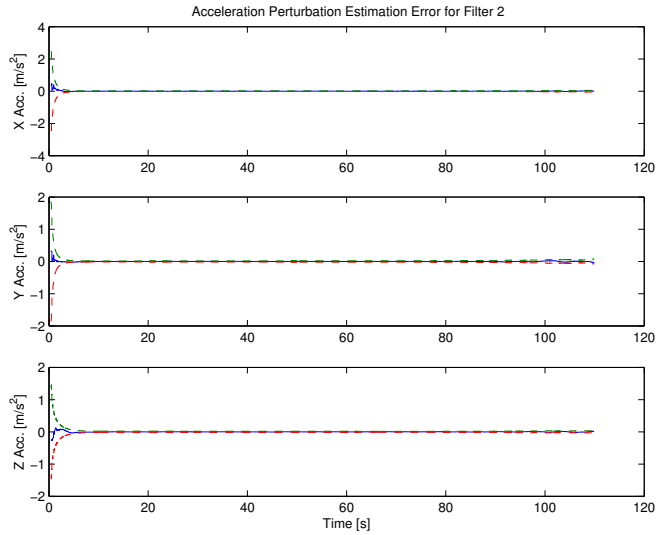

Figure 4.9: Decoy acceleration estimation errors for $\beta_m = 4 \times 10^4 \ Kg/m^2$, averaged over 30 Monte Carlo runs

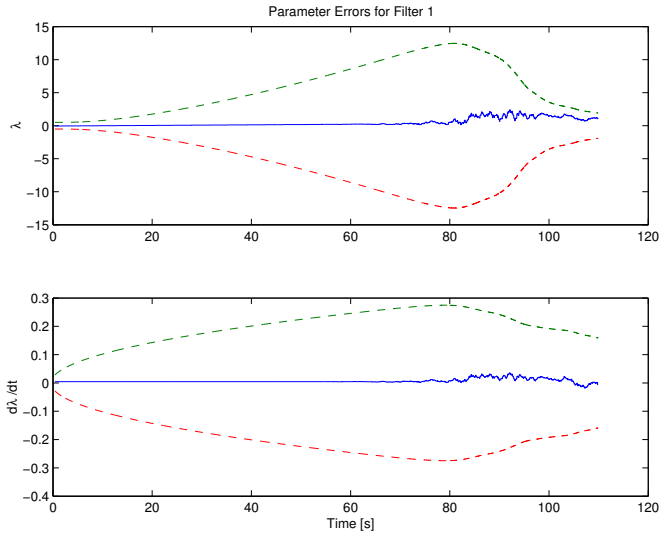

Figure 4.10: Target λ and $\dot{\lambda}$ estimation errors for $\beta_m = 4 \times 10^4 \ Kg/m^2$, averaged over 30 Monte Carlo runs

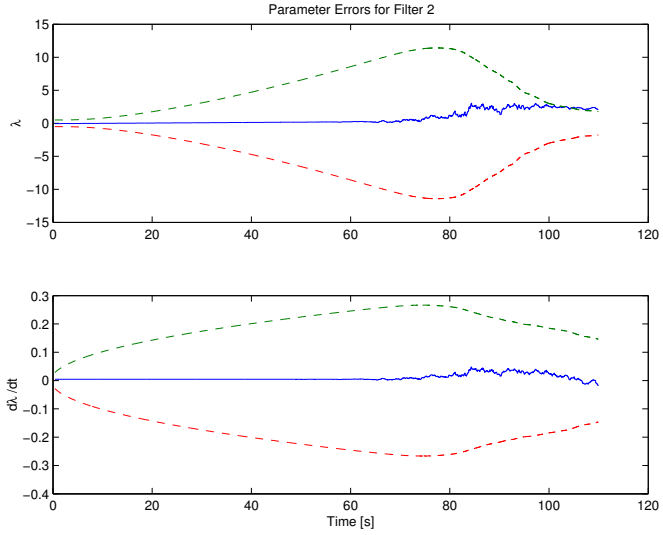

Figure 4.11: Decoy λ and $\dot{\lambda}$ estimation errors for $\beta_m = 4{\times}10^4 \ Kg/m^2$, averaged over 30 Monte Carlo runs

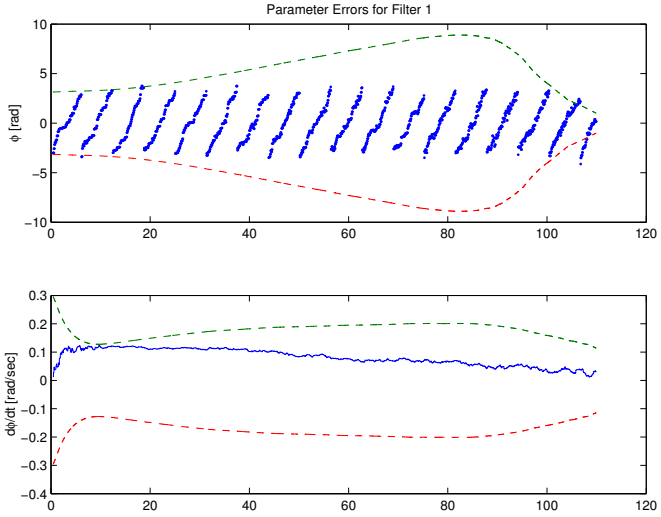

Figure 4.12: Target ϕ and $\dot{\phi}$ estimation errors for $\beta_m = 4 \times 10^4 \ Kg/m^2$, averaged over 30 Monte Carlo runs

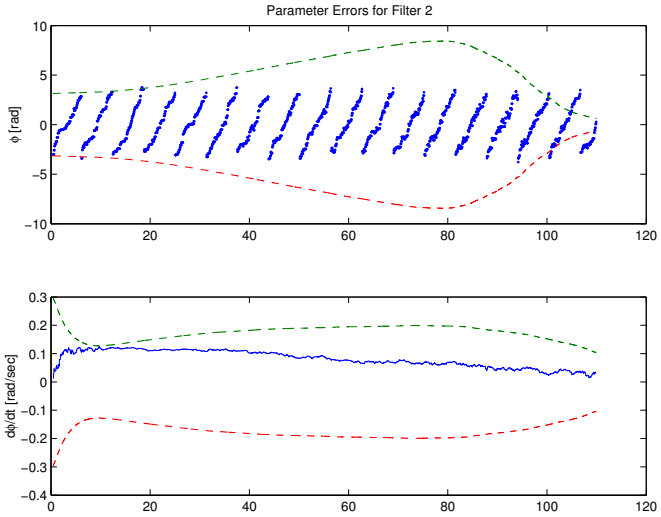

Figure 4.13: Decoy ϕ and $\dot{\phi}$ estimation errors for $\beta_m = 4 \times 10^4 \ Kg/m^2$, averaged over 30 Monte Carlo runs

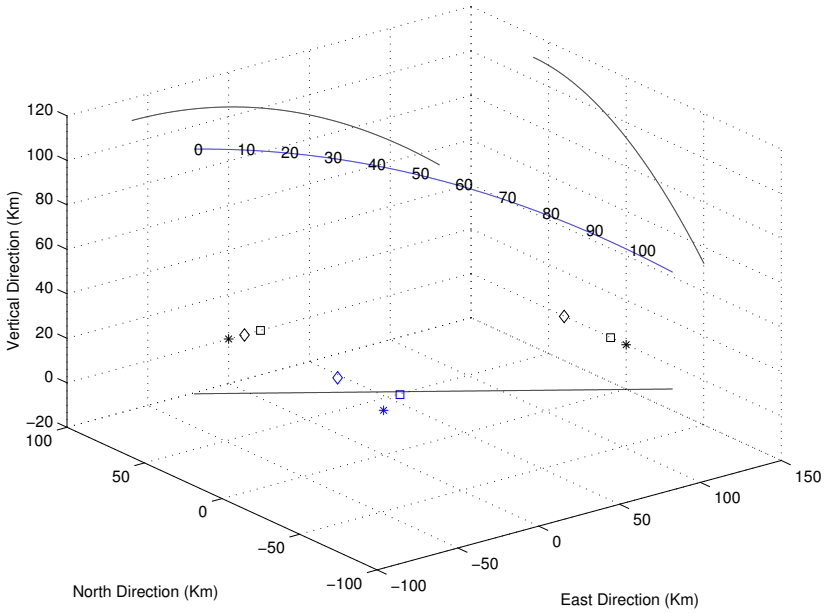

Figure 4.14: Trajectory with $\beta_m = 4 \times 10^3\ Kg/m^2$. Projections of the radars and the trajectory along each axis are in black.

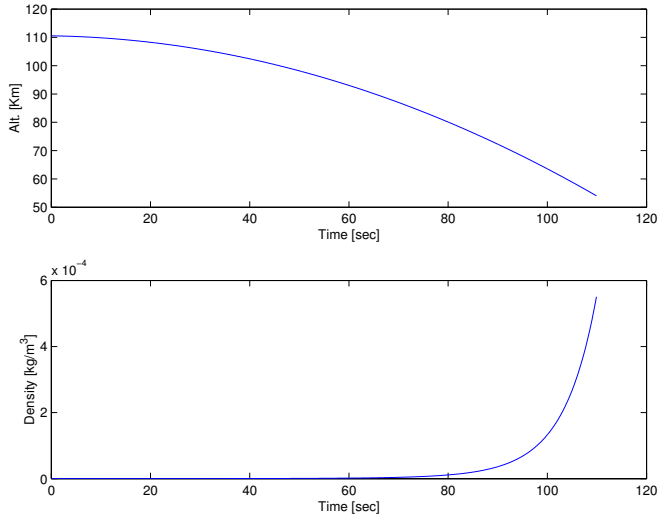

Figure 4.15: True altitude and surrounding atmospheric density for $\beta_m = 4 \times 10^3\ Kg/m^2$

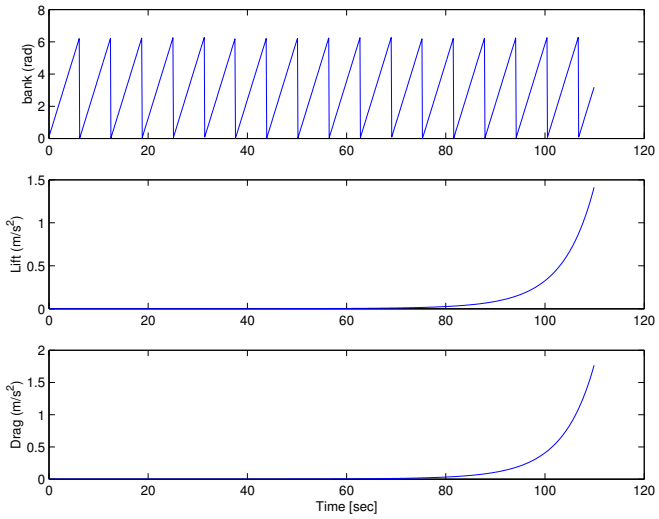

Figure 4.16: Bank angle, and lift and drag accelerations for $\beta_m = 4 \times 10^3\ Kg/m^2$

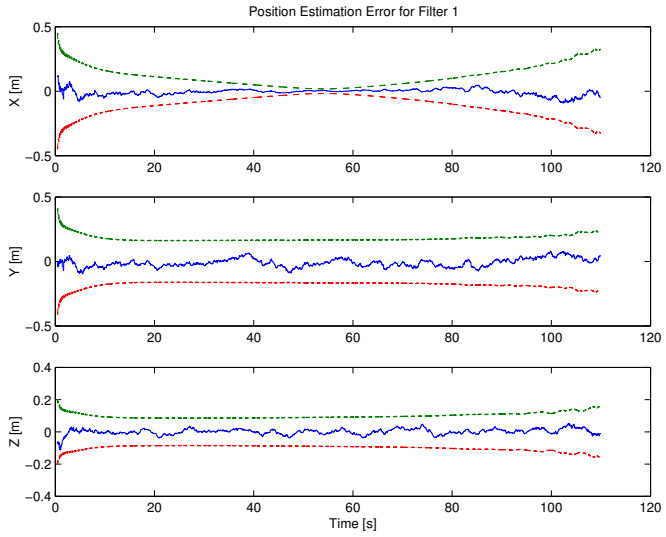

Figure 4.17: Target position estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs

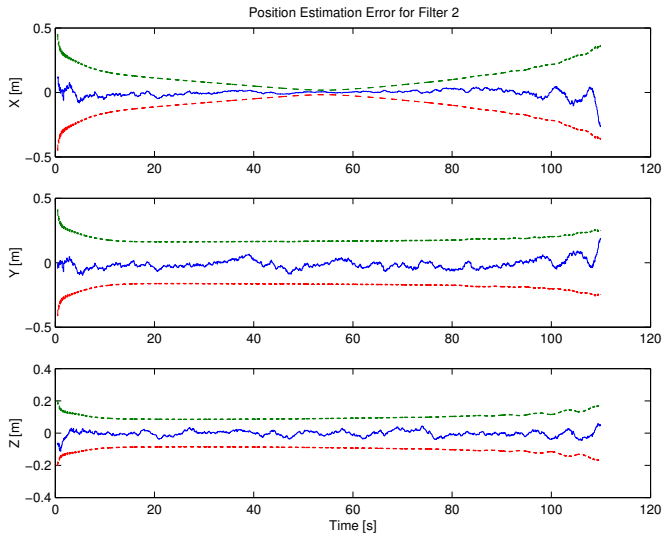

Figure 4.18: Decoy position estimation errors for $\beta_m = 4 \times 10^3 \ Kg/m^2$, averaged over 30 Monte Carlo runs

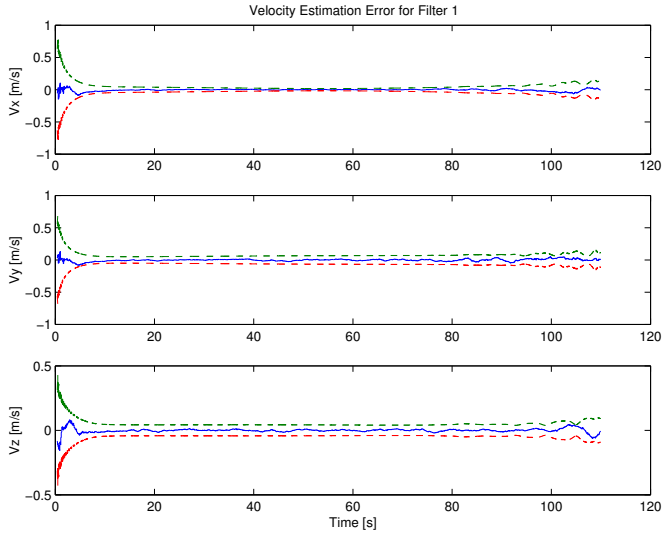

Figure 4.19: Target velocity estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs

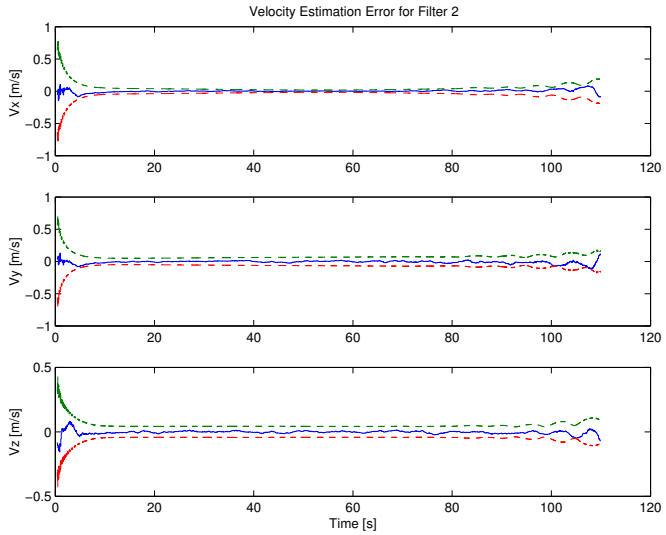

Figure 4.20: Decoy velocity estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs

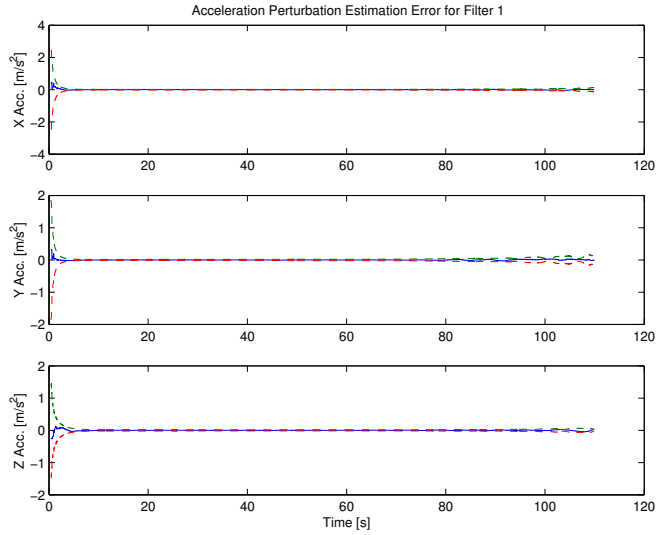

Figure 4.21: Target acceleration estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs

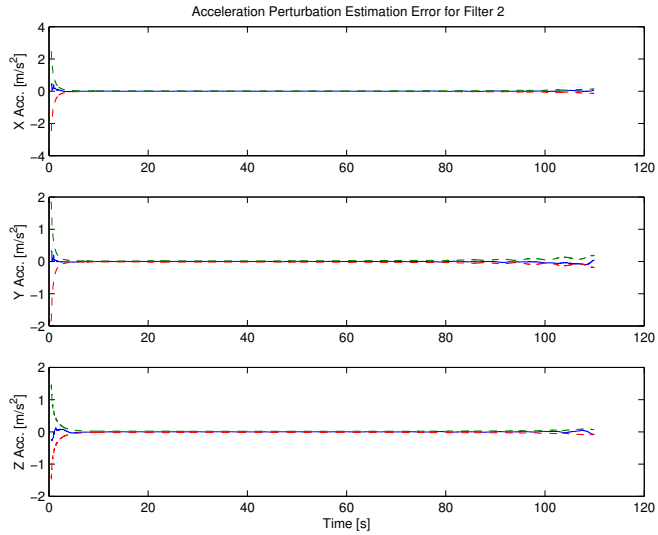

Figure 4.22: Decoy acceleration estimation errors for $\beta_m = 4 \times 10^3\ Kg/m^2$, averaged over 30 Monte Carlo runs

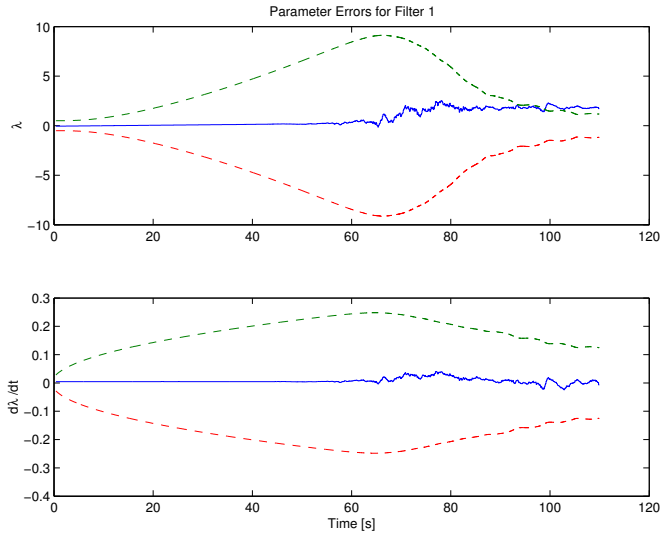

Figure 4.23: Target λ and $\dot{\lambda}$ estimation errors for $\beta_m = 4 \times 10^3 \ Kg/m^2$, averaged over 30 Monte Carlo runs

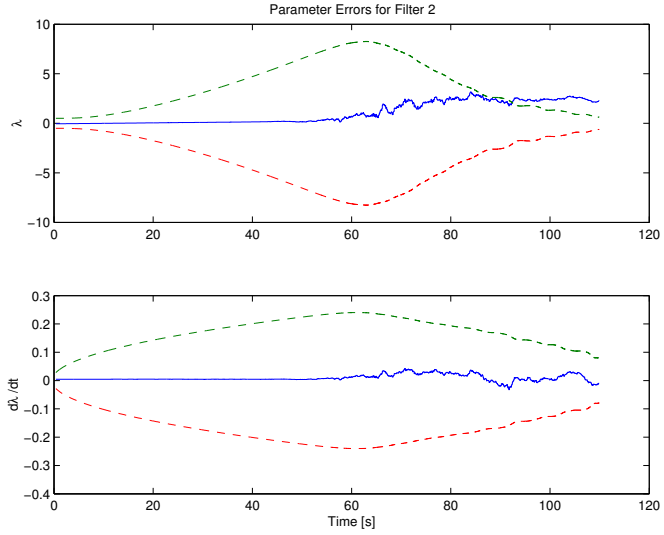

Figure 4.24: Decoy λ and $\dot{\lambda}$ estimation errors for $\beta_m = 4{\times}10^3 \ Kg/m^2$, averaged over 30 Monte Carlo runs

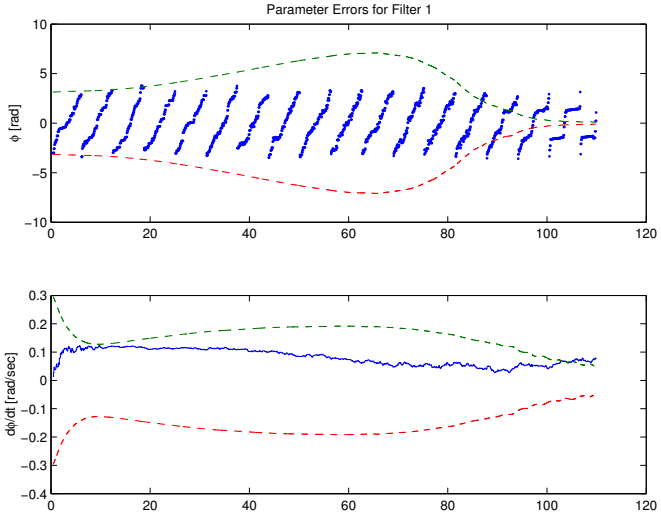

Figure 4.25: Target ϕ and $\dot{\phi}$ estimation errors for $\beta_m = 4 \times 10^3 \ Kg/m^2$, averaged over 30 Monte Carlo runs

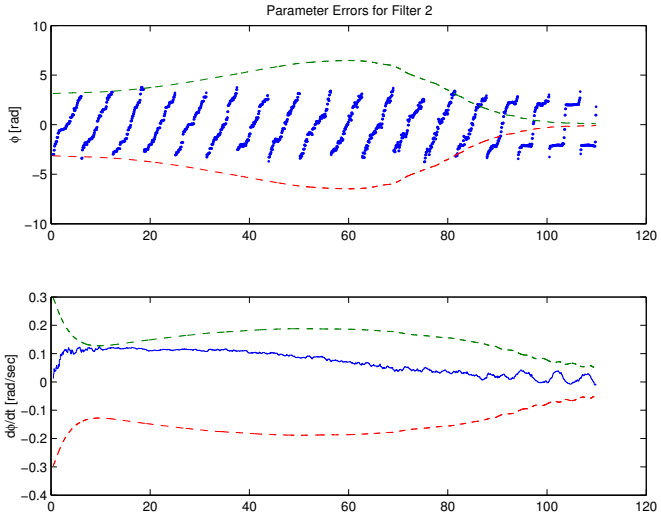

Figure 4.26: Decoy ϕ and $\dot{\phi}$ estimation errors for $\beta_m = 4 \times 10^3 \ Kg/m^2$, averaged over 30 Monte Carlo runs

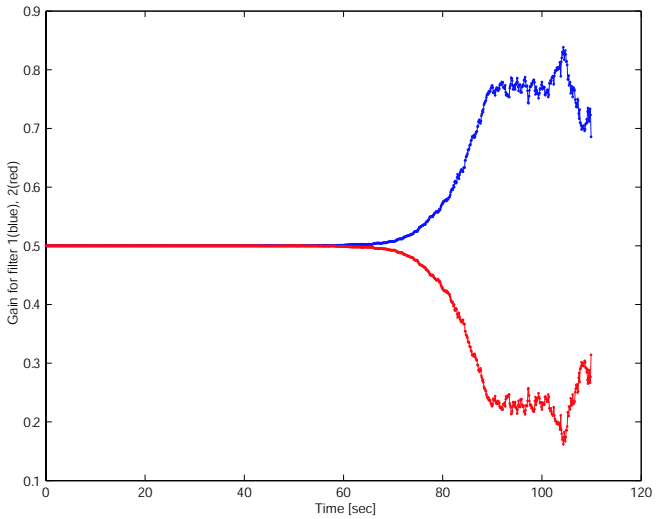

Figure 4.27: Gating network gain histories ($\beta_m = 4 \times 10^4\ Kg/m^2$): real parameter value (blue), 50% (red)

4.3.4 Conclusion

The extended Kalman filter was used to track reentering objects from a network of ground-based radars. A regulated mixture-of-experts fused the range, range rate and angle measurements to produce state estimates for two a priori values of the ballistic coefficient, one corresponding to a real target and the other one to a decoy. Simulations were made for two values of the real target ballistic coefficient. In both cases, the target was identified rapidly in a short time. Although the states in the Earth-centered frame were correctly tracked, it was found that some parameters of the lift and drag model are less observable, especially λ.

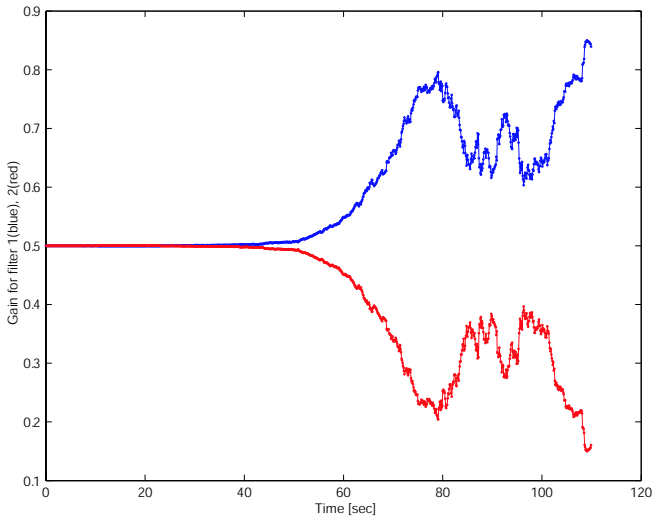

Figure 4.28: Gating network gain histories ($\beta_m = 4 \times 10^3\ Kg/m^2$): real parameter value (blue), 50% (red)

4.4 Torsion Analysis

The torsion is a measure of the motion of an object out of the osculating plane [33]. A zero torsion means that the object trajectory is contained in the plane. The larger the absolute value of the torsion, the more important the displacement of the osculating plane. Note that the torsion can be a negative number. The trajectory torsion was found to behave in a singular way that needed to be investigated. Explaining the shape of the torsion curve could also provide more insight into the dynamic of atmospheric reentry.

The effects of torsion become important at a low altitude, below the 50 Km limit of the target identification process. We can consider the tracking process as a two step sequence : first, the target is identify at high altitude, then within the remaining distance to the ground the target is tracked with increasing accuracy. The simulation in this section will therefore focus on the tracking of targets below 50 Km.

4.4.1 Torsion Analysis

In the following discussion, all scenarii will share the following parameters :

- $\beta_m = 4 \times 10^4 \ kg/m^2$
- $C_{D_0} = .033$ and $(\frac{C_L}{C_D})_{max} = 1$
- $\lambda_0 = 2$
- $\phi_0 = 0 \ rad.$ and $\dot{\phi} = 1 \ rad/sec$

In addition, for simplicity sake, the gravity field will be a central gravity (no J_2). The torsion formula,

$$\tau = \frac{\ddot{\mathbf{r}} * \dot{\mathbf{r}}}{\|\ddot{\mathbf{r}} * \dot{\mathbf{r}}\|} \odot \dddot{\mathbf{r}}, \tag{4.15}$$

, where $\odot$ represents an inner product, has units of $1/length.$

Recalling the triple time derivative of the position vector (see Eq. 3.18)

$$\begin{aligned} \dot{\boldsymbol{a}}_T &= \overbrace{\left[\omega^w * \left(\ddot{\mathbf{r}} + \frac{\mu}{r^3}\mathbf{r}\right)\right]}^{(a)} + \overbrace{\left[\dot{\phi}\mathbf{e}_1^w * \left(\ddot{\mathbf{r}} + \frac{\mu}{r^3}\mathbf{r}\right)\right]}^{(b)} - \overbrace{\left[\dot{D}\mathbf{e}_1^w\right]}^{(c)} \\ &+ \underbrace{\dot{L}\left[-\mathbf{e}_2^w sin\varphi + \mathbf{e}_3^w \cos\varphi\right]}_{(d)} - \underbrace{\frac{\mu}{r^3}\dot{r}\left[\mathbf{e}_1^w - 3(\frac{\mathbf{r}}{r} \odot \mathbf{e}_1^w)\frac{\mathbf{r}}{r}\right]}_{(e)}, \end{aligned} \tag{4.16}$$

we find that the terms (a) and (c), which correspond, respectively, to the in-plane maneuvering and the drag gradient of the MaRV model, do not contribute to the out-of-plane motion. After some manipulation, the resulting

expression for torsion is found to be

$$\begin{aligned}\tau \quad &= \overbrace{-\dot{\phi}\left[\frac{1}{\dot{r}} + \frac{\dot{r}}{\|\ddot{\mathbf{r}} * \dot{\mathbf{r}}\|^2}(\ddot{\mathbf{r}} - (\mathbf{e}_1^w \odot \ddot{\mathbf{r}})\mathbf{e}_1^w) \odot \frac{\mu}{r^3}\mathbf{r}\right]}^{\tau_b} \\ &\quad \overbrace{- \dot{L}\left[\frac{\dot{r}}{\|\ddot{\mathbf{r}} * \dot{\mathbf{r}}\|^2}(\mathbf{e}_3^w sin\varphi + \mathbf{e}_2^w \cos\varphi) \odot \mathbf{u_z}\right]}^{\tau_d} \\ &\quad \underbrace{- \frac{3\mu}{r^3}\frac{\dot{r}^2}{\|\ddot{\mathbf{r}} * \dot{\mathbf{r}}\|^2}(\frac{\mathbf{r}}{r} \odot \mathbf{e}_1^w)\left[(\frac{\mathbf{r}}{r} * \ddot{\mathbf{r}}) \odot \mathbf{e}_1^w\right]}_{\tau_e}.\end{aligned} \tag{4.17}$$

The three terms of the torsion are τ_b, which is related to the rotation rate $\dot{\phi}$ of the lift vector, τ_d, which is related to the change in magnitude of the lift vector, and τ_e, which is related to changes in the gravitational field. Typically, the latter two are negligible with respect to τ_b, as illustrated in Figure 4.29. As further example, Fig. 4.30 shows the trajectory resulting from setting $\dot{\phi} = 0$: the vehicle path is limited to a plane. From now on, we will consider τ_b synonymous with τ.

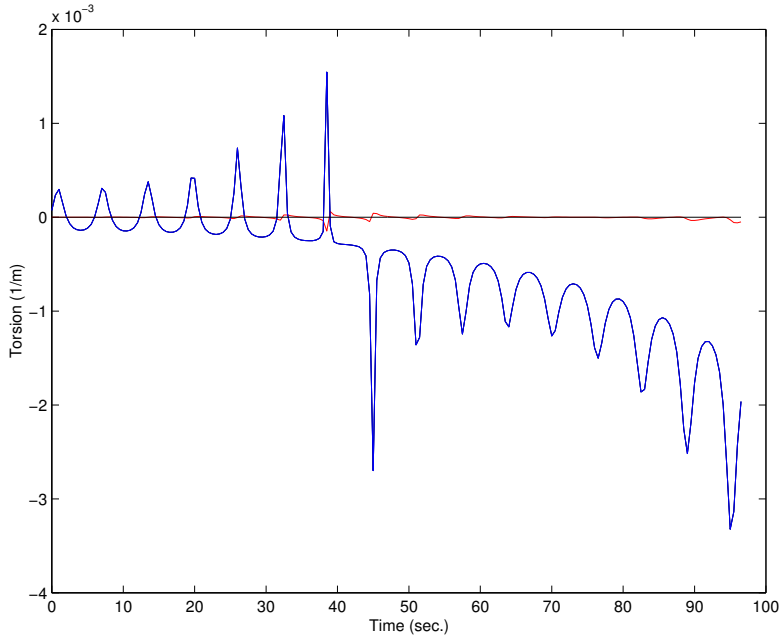

Figure 4.29: Torsion terms with $\beta_m = 4.10^4$: τ_b (blue), τ_d (red) and τ_e (black).

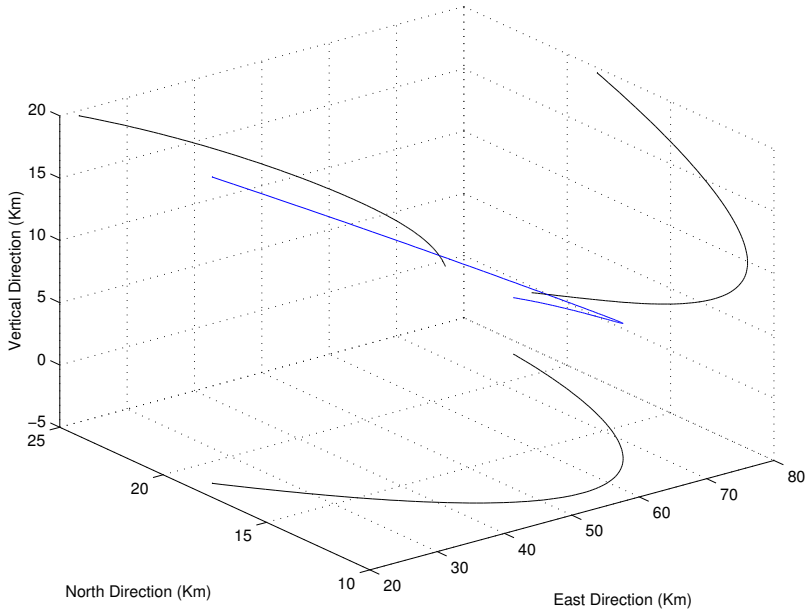

Figure 4.30: Trajectory with $\beta_m = 4.10^4$ and $\dot{\phi} = 0$.

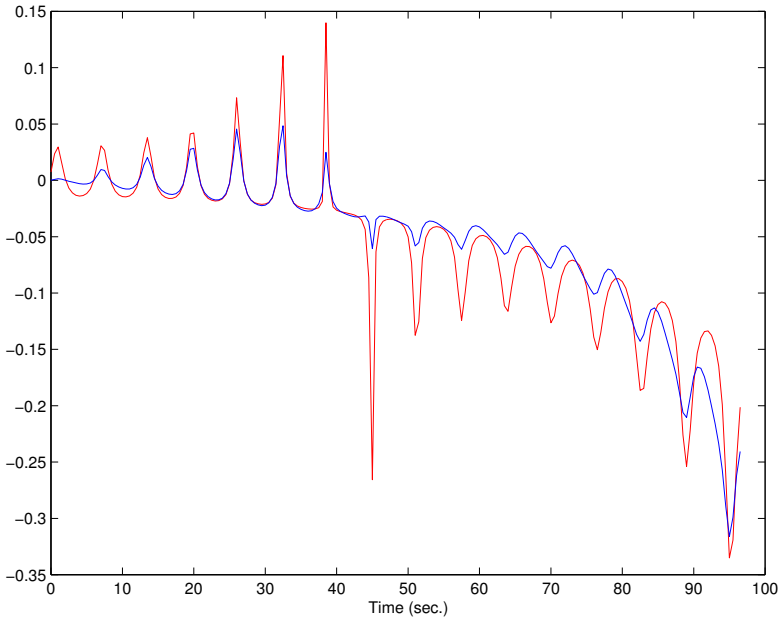

Figure 4.31: Cosine of the angle between ω and $\mathbf{F}$. The torsion ($\times 100$) has been superposed in red.

The ultimate cause of torsion, that is, out-of-plane motion, is that the sum of forces acting on the vehicle are steering it outside of the current osculating plane. One way to visualize this is to plot the angle between $\boldsymbol{\omega}$, normal to the osculating plane, and the vector of the resulting forces (lift, drag and weight) acting on the MaRV, $\boldsymbol{F}$, as we do in Fig. 4.31. The torsion coincides with the motion of $\boldsymbol{F}$ outside of the osculating plan.

Let us consider τ_b in Eq. 4.17. Two components in τ_b can be identified, and are plotted separately in Fig. 4.32. The first is a secular term given by

$$\tau_{b_1} = \frac{-\dot{\phi}}{\dot{r}}, \tag{4.18}$$

which is proportional, but of opposite sign, to the rotation rate of the lift vector, and inversely proportional to the magnitude of the velocity: as it falls

deeper into the atmosphere, the MaRV slows downs and it becomes easier to move out of the motion plane. Hence, the torsion naturally increases. The second term is a pseudo-periodic term given by

$$\tau_{b_2} = \frac{\dot{r}}{\|\ddot{\mathbf{r}} * \dot{\mathbf{r}}\|^2}(\ddot{\mathbf{r}} - (\mathbf{e}_1^w \odot \ddot{\mathbf{r}})\mathbf{e}_1^w) \odot \frac{\mu}{r^3}\mathbf{r}. \tag{4.19}$$

This term is more complex to analyze, and will be the main subject of the following discussion. First, there are peaks of torsion occurring at intervals equal to the rotation period of the lift vector. Those peaks occur when the lift and gravity vectors are in opposing direction as illustrated in Fig. 4.33. Second, there is a change of sign of τ_{b_2} midway through the trajectory. The sign of τ_{b_2} is the same as the one of τ_{sign}, defined as

$$\tau_{sign} = (-\ddot{\mathbf{r}} + (\mathbf{e}_1^w \odot \ddot{\mathbf{r}})\mathbf{e}_1^w) \odot \frac{\mu}{r^3}\mathbf{r}. \tag{4.20}$$

We can rewrite τ_{sign} as

$$\tau_{sign} = g^2 + \mathbf{L} \odot \mathbf{g} - \|\mathbf{e}_1^w \odot \mathbf{g}\|^2 . \tag{4.21}$$

Therefore, τ_{sign} is positive if

$$\begin{aligned} g^2 + \mathbf{L} \odot \mathbf{g} - \|\mathbf{e}_1^w \odot \mathbf{g}\|^2 &> 0 \\ \text{with} \\ 1 + \frac{\mathbf{L}}{g} \odot \frac{\mathbf{g}}{g} - \left\|\mathbf{e}_1^w \odot \frac{\mathbf{g}}{g}\right\|^2 &> 0. \end{aligned} \tag{4.22}$$

Define α as the angle between $\mathbf{e}_1^w$ and $\mathbf{g}$, and β as the angle between $\mathbf{L}$ and $\mathbf{g}$. Then, it follows that τ_{sign} is positive if

$$sin^2\alpha + \frac{L}{g}\cos\beta > 0. \tag{4.23}$$

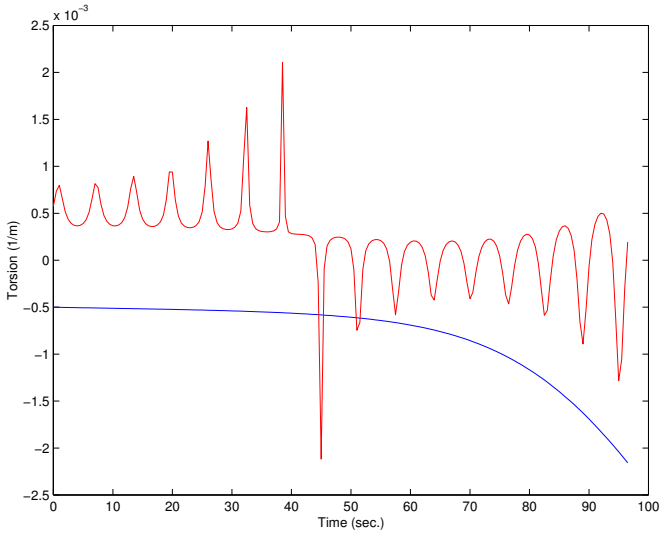

Figure 4.32: Torsion terms with $\beta_m = 4 \times 10^4\ Kg/m^2$: τ_{b_1} (blue) and τ_{b_2} (red).

Now, early in the trajectory, at high altitude, $\sin\alpha$ is close to 1, since the initial flight path of the MaRV is just slightly below the horizontal (about $0 - 10\ deg.$) For the same reason $\cos\beta$ takes a value between -1 and 1. The sign of τ_{sign} depends then on the ratio L/g. While that ratio remains below 1, τ_{b_2} is always positive. Above 1, τ_{b_2} is periodically negative, as shown in Fig. 4.34.

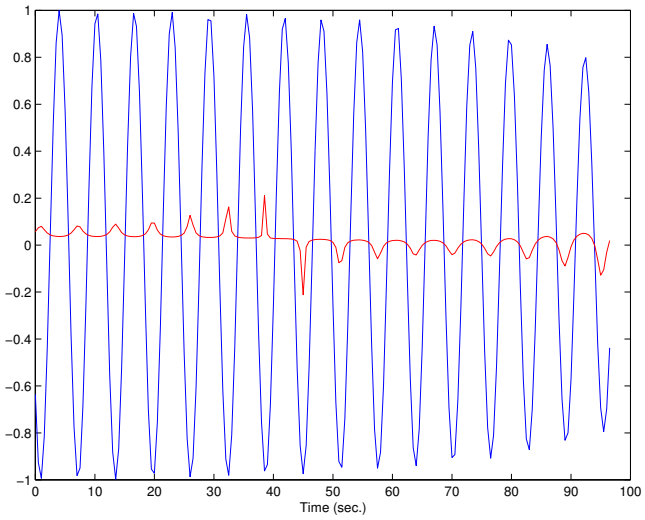

Figure 4.33: Cosine of the angle between the lift and gravity vectors. The torsion ($\times 100$) has been added in red.

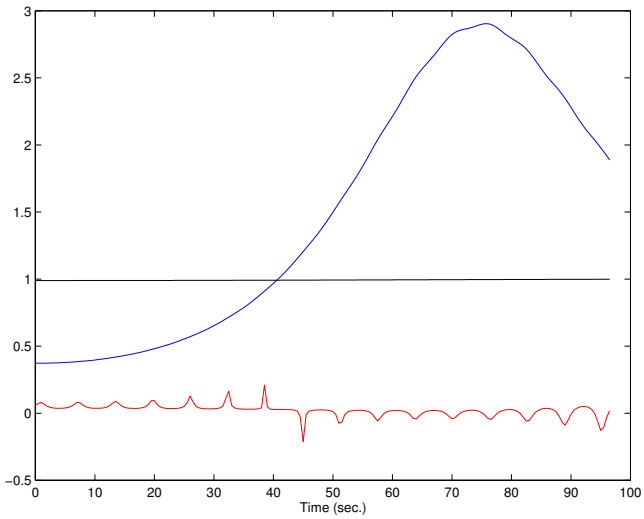

Figure 4.34: Lift (blue) and gravity (black), both in g's. τ_{b_2} added in red.

Overall, after adding the secular and the pseudo-periodic terms, the value of τ_b allows us to describe the motion of the MaRV. Initially after entering the atmosphere, the osculating plan of the trajectory begins to oscillate (τ_b is changing sign), with the strongest torsion occurring left of the flight path (τ_b positive). Once the lift exceeds the gravity force, the torsion remains negative and the vehicle starts spiralling to the right . In this scenario, lift remains greater than gravity to the end of the trajectory. This is not the case for instance with $\beta_m = 4.10^3\ Kg/m^2$, where lift decreases toward the end (Fig. 4.35). However, since α decreases (the MaRV is diving steeper toward the ground), τ_{sign} cannot become strictly positive anymore, and thus τ_b remains negative to the end.

4.4.2 Conclusion

Torsion of the MaRV's trajectory was analyzed and illustrated with a few scenarii. Torsion curves were explained in detail, and it has been shown

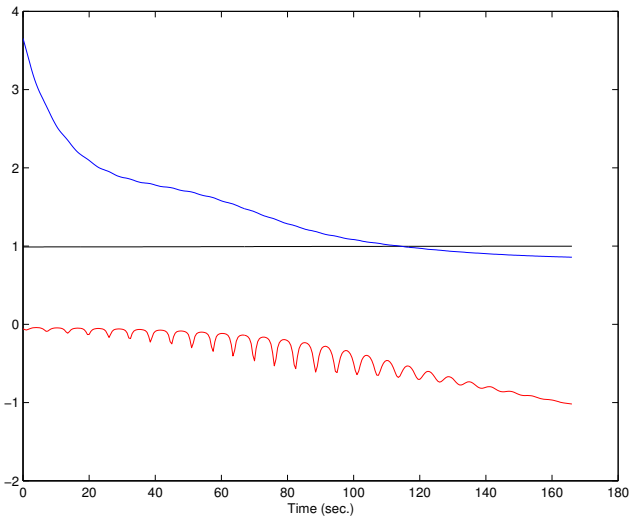

Figure 4.35: Lift (blue) and gravity (black), both in g's, for $\beta_m = 4 \times 10^3\ Kg/m^2$. $\tau_{b_2}(\times 100)$ added in red.

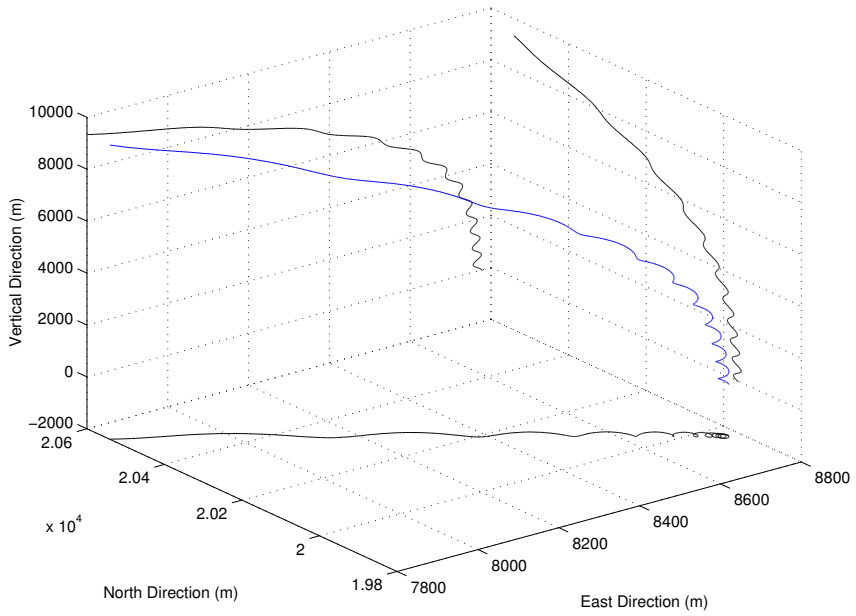

Figure 4.36: Detail of trajectory with $\beta_m = 4 \times 10^3\ Kg/m^2$

that torsion can be used to describe the overall shape of the MaRV's trajectory, like detecting a spiraling motion and its direction. Measuring torsion is then of great importance for guidance and tracking applications.

Chapter 5

Mars Entry Precision Navigation

5.1 Introduction

In this chapter, we apply the regulated mixture-of-experts to an on-board entry navigation system for a Mars lander. The measurements and the objectives are different from the tracking application in Chapter 4. However, the mixture-of-experts architecture permits us to work with different models of the real world in a parallel manner, and to make use of their relative strengths to achieve a better understanding in real time of the state of the vehicle and the external environment.

5.1.1 Rational for Mars Entry Precision Navigation

It is expected that future Mars lander missions, up to and including the Mars Sample Return Mission, will be able to focus on specific locations at the surface of the planet which, having been previously remotely sensed and mapped in detail from orbit, will have been found of particular geological interest. Such an area might be no more than a few kilometers wide, and the capacity for movement at the surface of the planet with a rover might be limited. Moreover, the region surrounding that area might be hazardous for landing, due to the presence of slope terrain, craters and boulders. Practical constraints require the use of an advanced navigation system to guide the vehicle to a pinpoint landing at the intended spot. Pinpoint landing is defined

to be less than 1 Km. Clearly, the guidance and control capability of the lander impacts the landing accuracy, that is, the ability to land at a specific point. Central to the guidance and control performance is the accuracy of the state estimate provided by the navigation system. This work presents an advanced navigation architecture capable of supporting precision landing.

The primary goal of all Mars landers to date has been to achieve a safe landing from Mars orbit (such as in the case of the Viking mission) or directly from their interplanetary approach trajectory (as with Mars Pathfinder). The expected landing point was allowed to be inside a wide ellipse on the surface (180×70 Km in the case of Pathfinder [34]). The landing uncertainty is due to the accumulation of several factors, such as the initial error in position and velocity just prior to atmospheric insertion, and the lack of knowledge in the atmospheric density profile, the winds, and knowledge of the true attitude of the spacecraft during the descent. The descent itself is comprised of several phases. First, the spacecraft, still contained within its aeroshell, makes a ballistic (or at best, a partially controlled) hypersonic descent during which only on-board inertial measurements, and possibly atmospheric measurements (such as stagnation point pressure), are available (ionization blackout would prevent any hypothetical ground guidance signal from reaching the lander, and Mars navigation satellites might not be available). Then, at an altitude of approximately 10 Km, one or several parachutes are deployed and the aeroshell is jettisoned, allowing ranging instruments on board to provide a measurement of the proximity to the ground. At this point, more advanced sensors can also map the terrain for collision avoidance. In the last few meters above the surface, the parachute is jettisoned and the craft lands on its own power. In the case of Mars Pathfinder, a landing bag system was employed which resulted

in a significant bouncing at the final phase. For precision landing, a powered terminal descent is required.

The first phase of the entry, being the most dynamically intensive and the poorest in available measurements, is the most important and most challenging in term of navigation. Precise knowledge of the position and velocity of the spacecraft is essential in order to reduce any initial state error and to guide it toward a terminal state from which the subsequent phases of the descent will lead to the intended landing point. A state estimation scheme (like the extended Kalman filter) should be capable of estimating the states of the lander by extracting as much information as possible from whatever measurements are available during the hypersonic descent, while also providing some measure of the remaining uncertainty surrounding these estimates. The state estimation should also be robust to temporary loss of data. But this assumes again that there is perfect knowledge of the environment and the dynamics involved, and most notably of the state of the martian atmosphere (density, temperatures, winds) in its upper region (down to 10 Km altitude). Current navigation methods employing dead-reckoning do not have satisfactory performance or robustness to meet future landing accuracy requirements.

We thus propose that a mixture-of-experts, spanning the space of possible atmospheric configurations, would be able to precisely navigate the aeroshell down to parachute deployment altitude, with error ellipsoids in position and velocity small enough as to be corrected during the subsequent phases of the descent. We will focus on the problem of estimating the density profile, which will be far more complex than an exponential model. The goal is not, of course, to span the entire realm of possible density profiles that might be encountered during the descent, since the mixture-of-experts is by nature discrete and will

contain a limited number of units. But unlike the tracking problem in Chapter 4, the goal here is not to test an hypothesis, but to obtain an optimal state estimate at any time. Also, unlike the previous case, the gating network cannot be left on its own until it reaches a steady state. For example, an expert can match the real atmosphere very well at low altitude, but be a poor fit at high altitude. If that filter is employed during the entire entry trajectory, the state estimate may diverge to such an extent that the filter will not be able to recover during the later part of the trajectory when it is, in theory, a correct model of the true environment. Thus each expert should be periodically reinitialized with the current best (highest gating weight) or, alternatively, optimal (weighted sum of all estimates) state estimate. How this is achieved will be explained later in this chapter.

5.1.2 Previous Examples of Atmospheric Entry Navigation with Conic Shell Spacecraft

No actual navigation system for atmospheric entry has ever used IMU data as "external" measurements employed in a Kalman filter to update the estimated states. The following is a brief review of entry navigation techniques used to date for conic shell spacecraft, similar in shape to the current and planned Mars landers.

Historically, the first of those is the Mercury spacecraft. There was, however, no guidance following the deorbit maneuver [35], and the spacecraft was rolled to minimize landing dispersions. The Gemini spacecraft had a built-in center of mass offset which allowed for some lifting capacity. Above $400,000$ feet, the guidance algorithm was provided with an estimated drag deceleration based on a stored air density profile. At lower altitude an IMU,

supplemented by an horizon sensor, provided the navigation data. In the Apollo spacecraft ([36]), aerodynamic acceleration measurements came from PIPAs (Pulsed Integrating Pendulous Accelerometers) as velocity increments, and were used directly as inputs in an *average-g* integration method along with a J_2 gravity model to propagate the position and velocity vectors.

To date, no Mars lander has been designed for precision landing. The Viking spacecraft landed with a number of measurement devices, notably an inertial reference unit (IRU) with three-axis accelerometers and gyros, redundant accelerometers and gyros, upper-atmosphere instruments (retarding potential analyser and mass spectrometer), and lower-atmosphere instruments (temperature probe and stagnation-pressure transducer) [37]. These instruments were mostly used, however, for post-flight trajectory reconstruction. There was no active guidance, and the control was limited to trim to an angle of attack of $-11.1\ deg$ prior to atmospheric entry, and as soon as $0.05\ g's$ were sensed, the spacecraft was left to its natural trim attitude, with a constraint of $1\ deg/s$ in attitude rate change.

The Pathfinder lander had no inertial measurement unit, as it performed a direct ballistic entry. Accelerometers provided total accelerations which were used to estimate the time remaining to parachute deployment [34]. Prior to parachute deployment, the spin and the natural stability of the conic shell kept the spacecraft at a prescribed zero-angle of attack. Landers targeted for landing on other planets, such as Stardust ([38]), have had similarly no active guidance between atmosphere interface and parachute deployment for precision landing.

5.1.3 Chapter Plan

The next section in this chapter describes the simulation used to validate the proof of concept for precision navigation with the regulated filter bank. We will explain how the true (or reference trajectories) were obtained along with the sequences of IMU measurements. The assumption leading to the definition of the vehicle model will be presented, and the important problem of modeling the atmosphere will be treated. Finally, we will examine the structure of the filter bank. Results of simulations with exponential and realistic MarsGRAM atmospheres will be presented.

5.2 Simulation Description

The simulation of the navigation filter will be based on a scenario for a 2009 Mars lander, which will use the Apollo guidance system for guidance during the reentry phase. The new navigation system will be tested in an open loop scenario and will not drive the guidance system directly. The navigation system state estimates will be compared to the actual states and to the estimate obtained through dead-reckoning. The test itself will be the average of many Monte Carlo runs with random variations in initial state estimate errors and measurement noise sequences. Reference trajectory data and actual measurements were generated by SORT, as described below. The simulation itself was written in MATLAB.

5.2.1 Generation of Reference Trajectories with SORT

The Simulation and Optimization of Rocket Trajectories (SORT), developed by the NASA Johnson Space Center, is a 3-degree of freedom (with

simulation of attitude motion through a digital autopilot), flight dynamics simulation software. It was initially developed for the Space Shuttle, but is adaptable to the Mars entry vehicle. SORT models environment factors, such as gravity and various atmosphere models, and key vehicle functions such as propulsion and guidance. It allows for the generation of discrete events, such as staging, and also includes targeting and optimization routines.

In this study, the true trajectory was obtained from a simulation of a September 2010 landing from atmospheric entry to parachute deployment. The state vector is composed of inertial position (Fig. 5.1), velocity (Fig. 5.2)and non-gravitational acceleration (Fig. 5.3). The non-gravitational accelerations are used, along with the attitude quaternions, to generate on-board, vehicle frame acceleration and gyroscope measurements. It was assumed that the gyroscope data are processed independently and that the attitude of the spacecraft in the inertial frame is known precisely. The attitude estimation problem is considered as an independent problem, and is left for a future study. Accelerometers, however, give noisy measurements, possibly further corrupted by various biases, scale factor errors and misalignments, which are all random constants [6]. As a first approach, the acceleration uncertainties will be limited in this dissertation to random measurement noise. It is understood that estimation of the biases and other errors is of importance in a realistic setting and will be the object of future studies. Other data of interest for the simulation provided by SORT, such as altitude, density, aerodynamic coefficients and bank angle, are plotted in Figures 5.4 through 5.6.

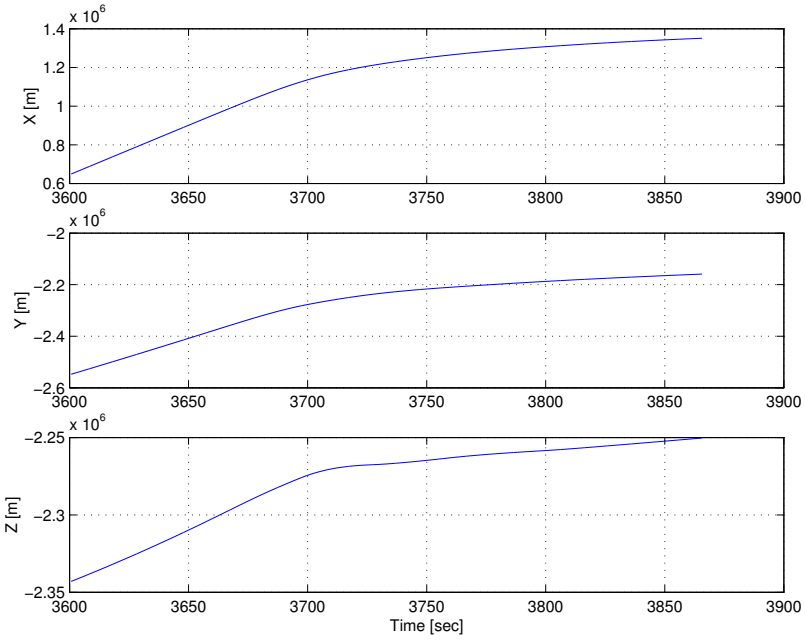

Figure 5.1: True MCI position

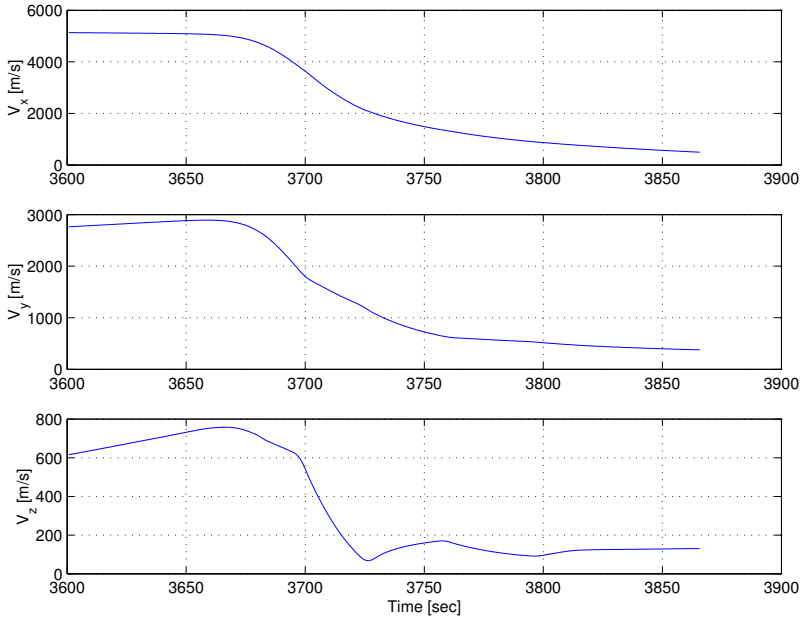

Figure 5.2: True MCI velocity

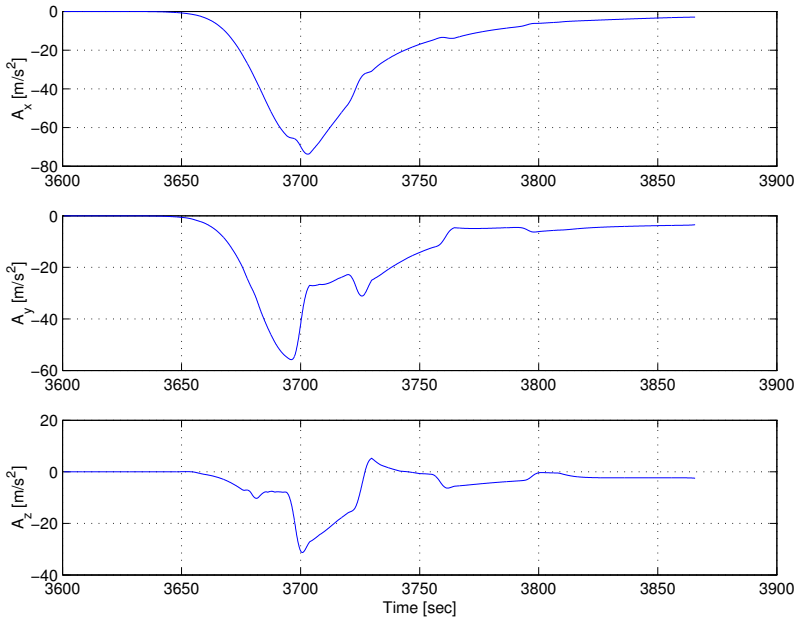

Figure 5.3: True MCI non-gravitational acceleration

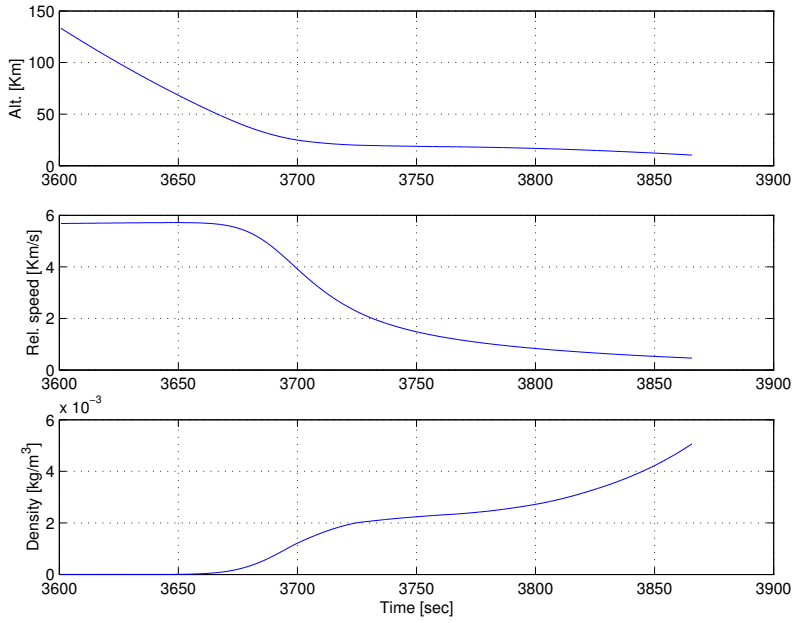

Figure 5.4: True altitude, relative velocity and atmospheric density

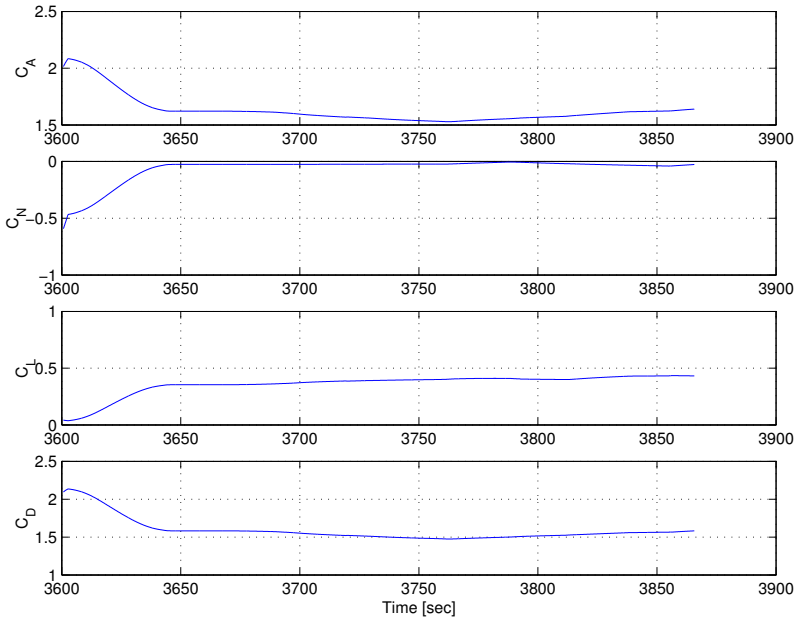

Figure 5.5: True aerodynamic coefficients

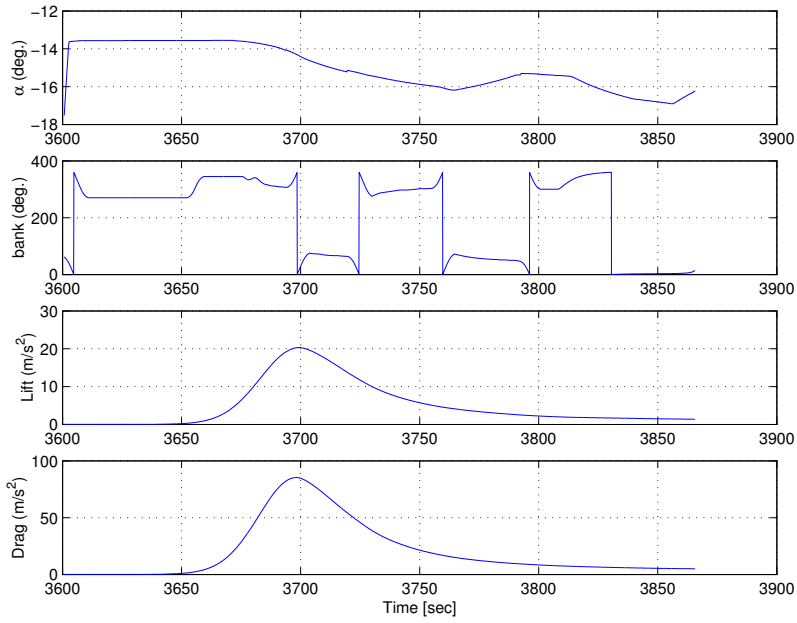

Figure 5.6: True maneuvering angles and wind-frame accelerations

Mass	Aerody. ref. surface	Diameter	L/D
2200 Kg	12.882 m^2	4.05 m	0.24

Table 5.1: Characteristics of the 2009 lander

5.2.2 Vehicle

The vehicle is a semi-conic aeroshell containing the lander. It is similar in shape to previous martian landers. The pitch orientation is maintained via vehicle asymmetry with an offset center of mass. Banking maneuvers are performed with a trim tab. Significant vehicle parameters are listed in Table 5.1.

5.2.3 Atmospheric Density Model

The SORT trajectory was generated with the Mars-GRAM model developed at NASA Marshall Space Filght Center [39]. It is a high-fidelity model of the martian atmosphere, taking into account both spatial and temporal variations. It utilizes data based on the NASA Ames Mars General Circulation Model (MGCM) and the University of Arizona Mars Thermospheric General Circulation Model (MTGCM), including variation of temperature, density, pressure and wind components with height, latitude, time of day and celestial longitude of the Sun. In addition, a parameter τ representing the optical depth of background dust level can be selected. The expected seasonal value of τ at arrival time was set to 0.45.

Time invariant approximate atmospheric models were considered for use in the filters, such as the COSPAR model and the interpolation of Viking data in Blanchard [40]. In fine, a simple two-layer exponential (the third layer is below the parachute deployment altitude) such as described in Tauber [1]

Altitude range (Km)	ρ_i (Kg/m^3)	β_i
> 36	0.03933	0.1181
9 to 36	0.01901	0.09804

Table 5.2: Atmospheric density profile constants (Reference trajectory from [1])

Filter	1	2	3	4	5	6
Coeff.	1	0.5	0.1	0.3	0.7	0.2

Table 5.3: Coefficients for each filter in the bank

has been used. The form of the approximate atmospheric density model is

$$\rho = \rho_i e^{-\beta_i y}, \tag{5.1}$$

where y is the altitude, and the constants are listed in Table 5.2. The constants in Table 5.2 were interpolated from Viking data, and do not by themselves make a suitable generic atmospheric model. However, they provide the basis upon which to generate a population of atmospheric models for the filter bank.

In order to populate the filter bank, both the parameters ρ_i and β_i can be varied. For the sake of simplicity, we will only vary ρ_i. Also, we will limit the size of the bank to six filters. They are obtained by multiplying ρ_i by the factors listed in Table 5.3.

The real density profile and those of the filter are plotted in Fig. 5.7 and 5.8 and the differences between them in Fig. 5.9 and 5.10. It is to be noted that the 5th filter (gray) is extremely close to the real atmosphere in the lower altitudes. This is helpful to test the filter bank, as the gating network should logically select this filter. In subsequent simulations, however, the coefficient for filter 5 will be changed so it is not as close to the real atmospheric density.

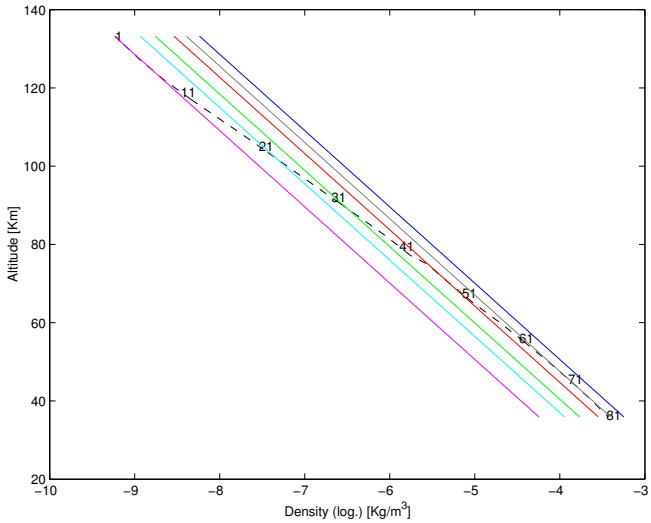

Figure 5.7: Actual (dotted line) and filter atmospheric density profiles down to 36 Km altitude (figures indicate time elapsed in seconds). Filter 1 : blue , 2 : red, 3 : magenta, 4 : green, 5 : gray, 6 : light blue.

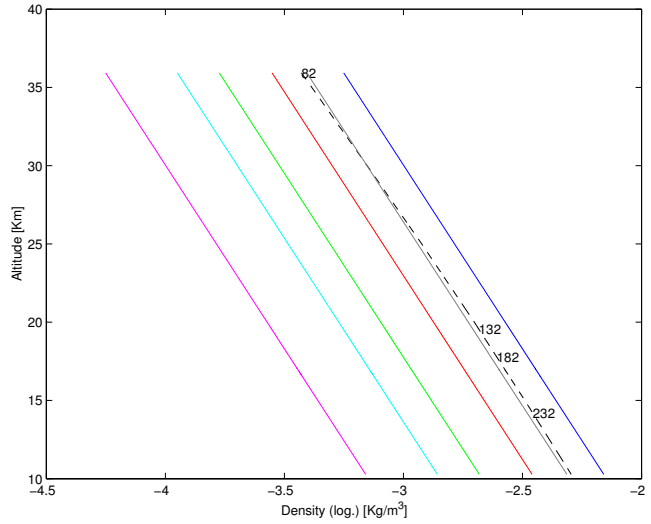

Figure 5.8: Actual (dotted line) and filter atmospheric density profiles down from 36 Km altitude down to parachute deployment. Filter 1 : blue , 2 : red, 3 : magenta, 4 : green, 5 : gray, 6 : light blue.

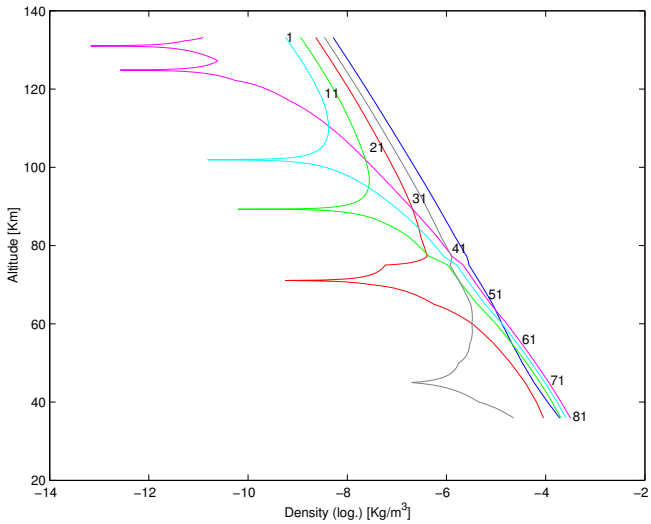

Figure 5.9: Differences between actual and filter atmospheric density profiles down to 36 Km altitude. Filter 1 : blue , 2 : red, 3 : magenta, 4 : green, 5 : gray, 6 : light blue.

5.2.4 Individual Filter Description

Each expert in the bank is a 9-states filter, with measurement update every 0.1 seconds. The gyroscopic angular rate is assumed known precisely and is integrated to provide quaternions of body frame orientation with respect to the Mars centered inertial frame. Acceleration measurements are corrupted with a random noise sequence. Both real and filter measurement noise are zero mean white sequences with a variance of $10^{-3} m/sec^2$, which is an upper bound on the white noise value for existing accelerometers.

The aerodynamic bank angle Φ is a control input, and $\dot{\Phi}$ (needed in the filter) is computed numerically. The aerodynamic coefficients C_D and C_L are set for the filter to constant values ($C_D = 1.55$ and $C_L = 0.35$). Although actual variations of these coefficients can be significant, as shown in Fig. 5.5, the estimation performance was found to not degrade significantly when using

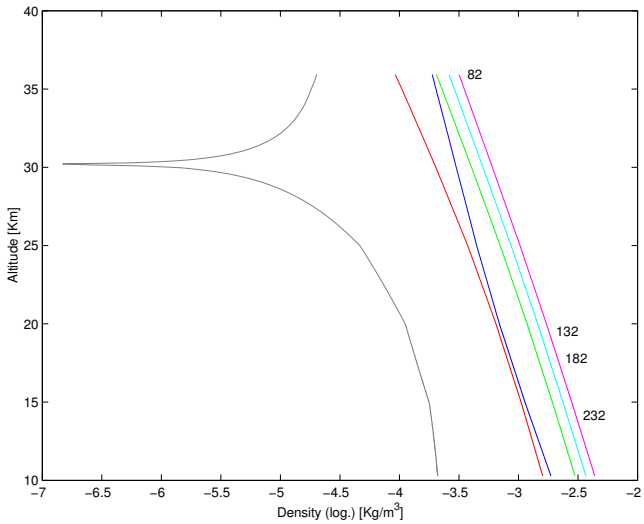

Figure 5.10: Differences between actual and filter atmospheric density profiles down from 36 Km altitude down to parachute deployment. Filter 1 : blue , 2 : red, 3 : magenta, 4 : green, 5 : gray, 6 : light blue.

a constant value. This is to be expected since C_D and C_L are constant during the most dynamic part of the entry, between $t = 3650sec$ and $t = 3750sec$ at the same time that the lift and drag (see Fig. 5.6) are at their peak values.

The process noise is applied to the acceleration states only. The values of the process noise are defined in the estimated RSW frame [1]. The reason is that the altitude, downrange and crossrange channels exhibit different estimation behavior. For example, the crossrange channel, as might be expected, is weakly observable. The process noise spectral density values were found through a filter tuning process. The squared non-zero diagonal elements of the process noise matrix expressed in the RSW frame are listed in Table 5.4.

[1]The RSW frame is composed of a unit vector along the spacecraft position vector ($\boldsymbol{R}$), a vector orthogonal to both the position and velocity vectors ($\boldsymbol{W}$), and a vector completing the triad ($\boldsymbol{S}$) roughly pointing in the direction of motion [41].

R	S	W
$10^{-4}\ m^2/s^5$	$10^{-4}\ m^2/s^5$	$10^{-3}\ m^2/s^5$

Table 5.4: Process noise spectral density values in the RSW frame

$\boldsymbol{r}$	$\boldsymbol{v}$	$\boldsymbol{a}$
$4000\ m$	$0.1\ m/s$	$0.1\ m/s^2$

Table 5.5: Initial estimation error covariance values

To account for the peak in acceleration during the entry trajectory, the process noise parameters are appropriately scaled by the norm of the current measured acceleration:

$$(1 + \|\boldsymbol{a}_{meas}\|)^2 \boldsymbol{Q}(t) \longrightarrow \boldsymbol{Q}(t) \tag{5.2}$$

The true initial state is

$$\begin{aligned} \boldsymbol{X}_{Pos} &= [6.487645 \times 10^5, -2.547429 \times 10^6, -2.34299 \times 10^6]m, \\ \boldsymbol{X}_{Vel} &= [5.13278 \times 10^3, 2.763147 \times 10^3, 6.154346 \times 10^2]m/s, \\ \boldsymbol{X}_{Acc} &= [-1.059417 \times 10^{-4}, -4.882649 \times 10^{-5}, ... \\ &\quad -1.31802 \times 10^{-5}]m/s^2. \end{aligned} \tag{5.3}$$

The initial state covariance matrix is diagonal, with the values listed in Table 5.5. These values also serve as the variance of the random initial state estimation errors at the beginning of each Monte Carlo run.

5.2.5 Filter Bank Description

The filter bank is a single layer of regulated experts, whose length is limited by the computation capability of the host computer. The following experiments employs six filters. All filters are initially equiprobable. The

internal activation weights u_i described in Equation 2.7 are all set to 10^{-2}. The learning rate is 1. This high value is necessitated by the rapid evolution in the objective performance of each filter.

The nature of the navigation system requires the filter bank to be periodically reinitialized (or restarted), with the current optimal state estimate (the weighted sum of all state estimates), so as to not later disadvantage the filters that perform poorly. It was found through trial and error that a restart every 20 seconds would allow enough time between restarts for the filters to develop distinct behavior. The state covariance needed at restart is more difficult to determine. Its derivation is presented next.

5.2.5.1 Fusion of Filter Covariances

Assume we have a bank of N filters, which we need to restart periodically with the optimal state estimate $\hat{\boldsymbol{x}}_{opt} = \sum_{i=1}^{N} g_i \hat{\boldsymbol{x}}_i$ (for the rest of this section, we will forgo the use of time indexes). In addition, a corresponding optimal covariance is required. The optimal covariance is defined as

$$\begin{aligned} \boldsymbol{P}_{opt} &= E[(\boldsymbol{x} - \hat{\boldsymbol{x}}_{opt})(\boldsymbol{x} - \hat{\boldsymbol{x}}_{opt})^T] \\ &= E[\tilde{\boldsymbol{x}}_{opt}\tilde{\boldsymbol{x}}_{opt}^T] \end{aligned} \tag{5.4}$$

where $\tilde{\boldsymbol{x}}_{opt} = \boldsymbol{x} - \hat{\boldsymbol{x}}_{opt}$. From the discussion in Chapter 2, the optimal estimate is a weighted sum of the individual state estimates, hence we have

$$\boldsymbol{P}_{opt} = E[(\sum_{i=1}^{N} g_i \tilde{\boldsymbol{x}}_i)(\sum_{i=1}^{N} g_i \tilde{\boldsymbol{x}}_i)^T]$$

and since the g_i are scalar

$$\begin{aligned} \boldsymbol{P}_{opt} &= \sum_{i,j=1}^{i,j=N} g_i g_j E[\tilde{\boldsymbol{x}}_i \tilde{\boldsymbol{x}}_j^T] \\ &= \sum_{i=1}^{i=N} g_i^2 \boldsymbol{P}_i + \sum_{i \neq j} g_i g_j \boldsymbol{P}_{ij} \\ &= \sum_{i=1}^{i=N} g_i^2 \boldsymbol{P}_i + \sum_{i=1}^{i=N-1} \sum_{i<j} g_i g_j (\boldsymbol{P}_{ij} + \boldsymbol{P}_{ij}^T) \end{aligned} \tag{5.5}$$

where $\boldsymbol{P}_{ij} = E[\tilde{\boldsymbol{x}}_i \tilde{\boldsymbol{x}}_j^T]$, for $i \neq j$.

The second term of Eq. 5.5 contains the cross-filter covariances $\boldsymbol{P}_{ij}$, which cannot be readily computed and, in fact, must be updated and propagated in a similar way to the covariances $\boldsymbol{P}_i$. There are $\frac{N!}{(N-2)!}$ cross-covariances possible, which is a significant computational burden. It is possible, however, to use an upper bound covariance instead. This can be computed as follows. First, the quantity

$$\boldsymbol{U}_{ij} = E[(\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j)(\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j)^T], \tag{5.6}$$

defined for $i \neq j$, is positive definite, that is

$$E[(\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j)(\tilde{\boldsymbol{x}}_i - \tilde{\boldsymbol{x}}_j)^T] \;>\; 0. \tag{5.7}$$

Expanding Eq. 5.7 yields

$$\boldsymbol{P}_i + \boldsymbol{P}_j - \boldsymbol{P}_{ij} - \boldsymbol{P}_{ij}^T \;>\; 0$$

or

$$\boldsymbol{P}_i + \boldsymbol{P}_j \;>\; \boldsymbol{P}_{ij} + \boldsymbol{P}_{ij}^T. \tag{5.8}$$

Substituting Eq. 5.8 into Eq. 5.5 yields the inequality

$$\boldsymbol{P}_{opt} \;<\; \sum_{i=1}^{i=N} g_i^2 \boldsymbol{P}_i + \sum_{i=1}^{i=N-1} \sum_{i<j} g_i g_j (\boldsymbol{P}_i + \boldsymbol{P}_j) = \boldsymbol{P}_{subopt} \tag{5.9}$$

where $\boldsymbol{P}_{subopt}$ is the suboptimal covariance to be used in the filter bank. Expanding Eq. 5.9 yields

$$\boldsymbol{P}_{subopt} \;=\; \sum_{i=1}^{i=N} g_i^2 \boldsymbol{P}_i + \sum_{i=1}^{i=N-1} \sum_{i<j} g_i g_j \boldsymbol{P}_i + \sum_{i=1}^{i=N-1} \sum_{i<j} g_i g_j \boldsymbol{P}_j.$$

Summing along j instead of i in the third term yields

$$\boldsymbol{P}_{subopt} \;=\; \sum_{i=1}^{i=N} g_i^2 \boldsymbol{P}_i + \sum_{i=1}^{i=N-1} \sum_{i<j} g_i g_j \boldsymbol{P}_i + \sum_{j=2}^{j=N} \sum_{i<j} g_i g_j \boldsymbol{P}_j.$$

Finally, switching i and j in the third term yields

$$\begin{aligned} \boldsymbol{P}_{subopt} &= \sum_{i=1}^{i=N} g_i^2 \boldsymbol{P}_i + \sum_{i=1}^{i=N-1} \sum_{i<j} g_i g_j \boldsymbol{P}_i + \sum_{i=2}^{i=N} \sum_{j<i} g_i g_j \boldsymbol{P}_i \\ &= \sum_{i=1}^{i=N} g_i \Big(\sum_{j=1}^{j=N} g_j\Big) \boldsymbol{P}_i \\ &= \sum_{i=1}^{i=N} g_i \boldsymbol{P}_i. \end{aligned} \tag{5.10}$$

So, we can use $\boldsymbol{P}_{opt} \simeq \boldsymbol{P}_{subopt} = \sum_{i=1}^{i=N} g_i \boldsymbol{P}_i$, and understand that it is an upper bound of the actual coordinates. This is acceptable because it reduces the computational burden while not leading to an overly optimistic filter bank.

5.3 Simulation Results

In this section, we test the filter bank. Each simulation is a series of 30 Monte Carlo runs, with random initial state errors and measurement

noise factors. The state estimate errors and other values of interest here are averaged over the Monte Carlo runs. The results are presented as follows: first, the evolution of the gating weight is shown. The state estimation errors are plotted along with the square root of the corresponding diagonal elements of the state estimation error covariance matrix. Final norms of the position and velocity errors will also be listed. Additional variables, computed from the estimated state before each propagation step, are also plotted. These include drag, lift and gravity acceleration errors, atmospheric density absolute and relative estimation errors, and altitude and relative velocity estimation errors.

5.3.1 First Filter Bank

With the filter models as described in the previous section, the total estimation error in position at the end of the navigation run is 1.389 Km, and 0.498 m/sec for the velocity estimation error. The corresponding plots are Figures 5.11- 5.16. In Fig. 5.11, we can see that the gating network, after 50 seconds of learning, chooses clearly as best filter number 5 (gray), which is indeed the one with the most accurate atmospheric model. Other models were closer higher in the trajectory, but were not selected, (except, for a short time, number 2 (red)). Note also the visible restarts in Fig. 5.14 every 20 seconds.

We do observe a sharp drop in the estimation performance at the very end of the simulation time, just before parachute deployment. It is especially visible in the estimated altitude and density, as shown in Fig. 5.16. It also leads filter 5 to lose weight to the benefit of the other filters (again, Fig. 5.11). The explanation for this behavior can be found in the filter density plot in Fig. 5.10. As the spacecraft descends, the absolute value of the filter density error grows (even for the best filter) and the estimation errors become large

enough across the filter bank to prevent the filter from making a selection. Future implementations of the filter bank should take this phenomenon into account by having some of their filters expert in the low altitude range (that is, just above 10 Km), and having enough of these, or just a few if the possible variations of the actual density profile can be narrow down, to provide a very close fit of the actual density.

For a global view of the performance of the mixture-of-experts and the resulting evolution in the position error covariance, the trajectory is plotted in three dimensions above a section of the martian surface in Fig. 5.17 and Fig 5.18. The covariance is represented, every $10sec$, by an ellipsoid whose dimensions are the singular values of the covariance matrix. Initially spherical, the ellipsoid is shaped by the information gathered by the mixture-of-experts. For example, from very early on, it is progressively flattened in the direction of motion, since the acceleration measured is for the most part the drag acceleration.

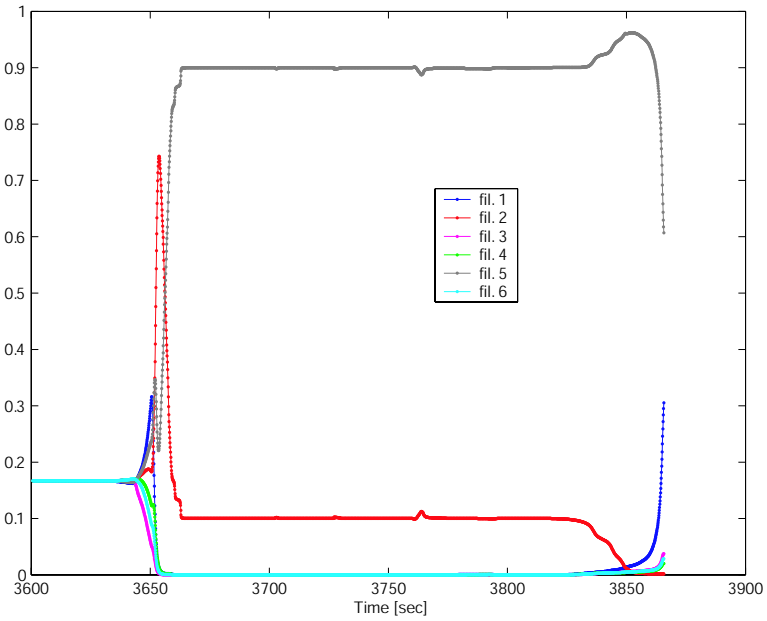

Figure 5.11: First filter bank: gating network weights

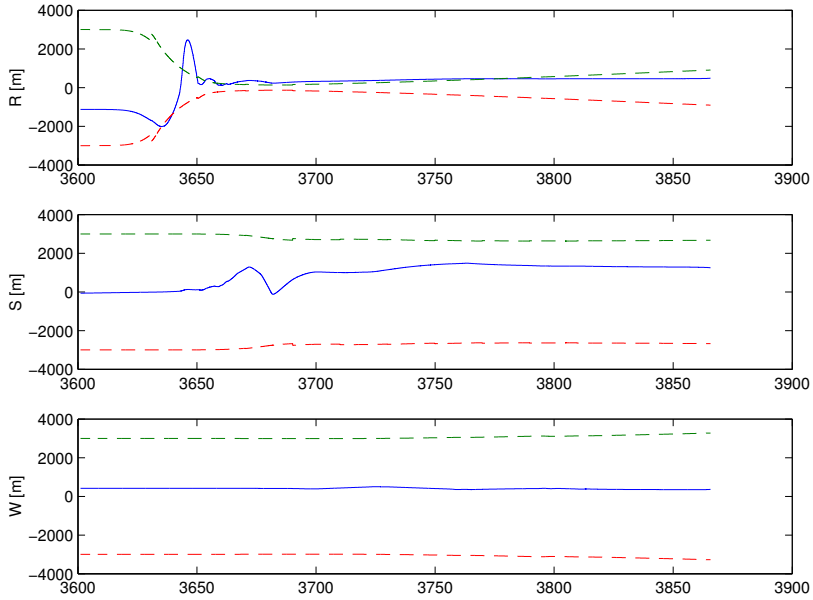

Figure 5.12: First filter bank: optimal RSW position estimation, averaged over 30 Monte Carlo runs

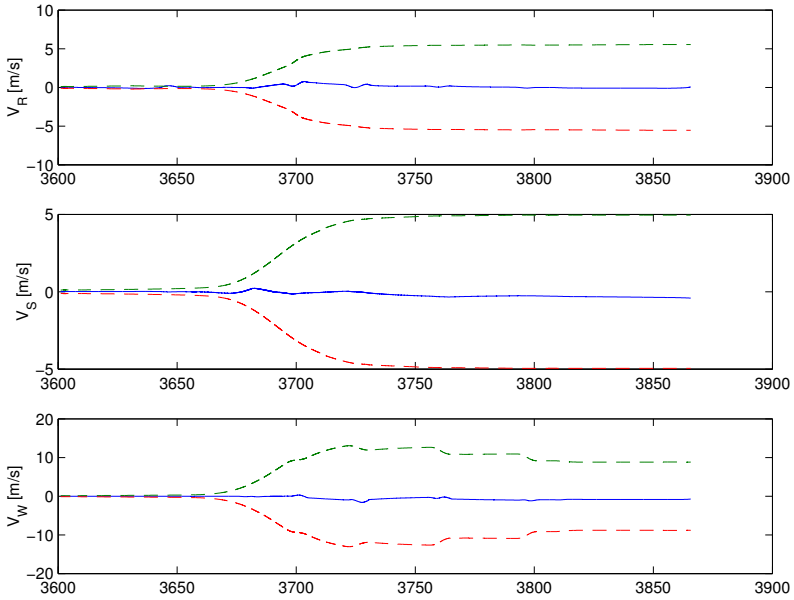

Figure 5.13: First filter bank: optimal RSW velocity estimation

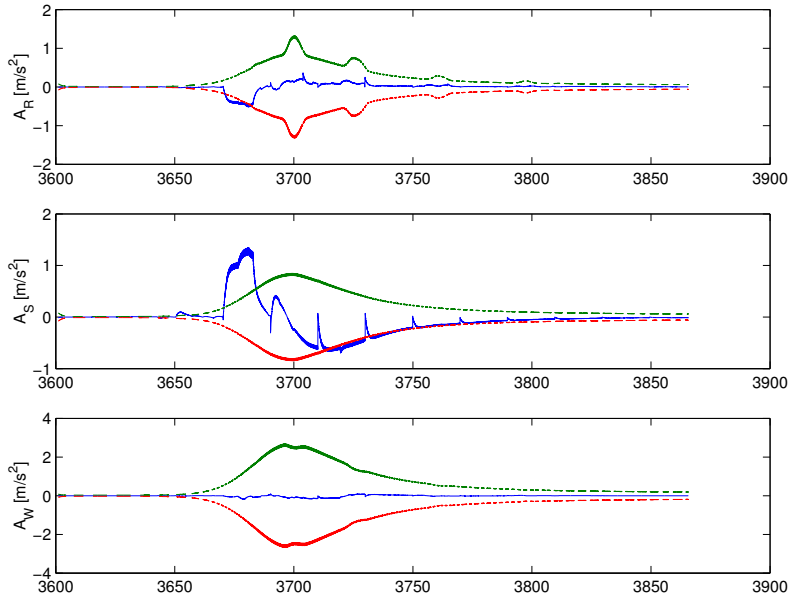

Figure 5.14: First filter bank: optimal RSW acceleration estimation

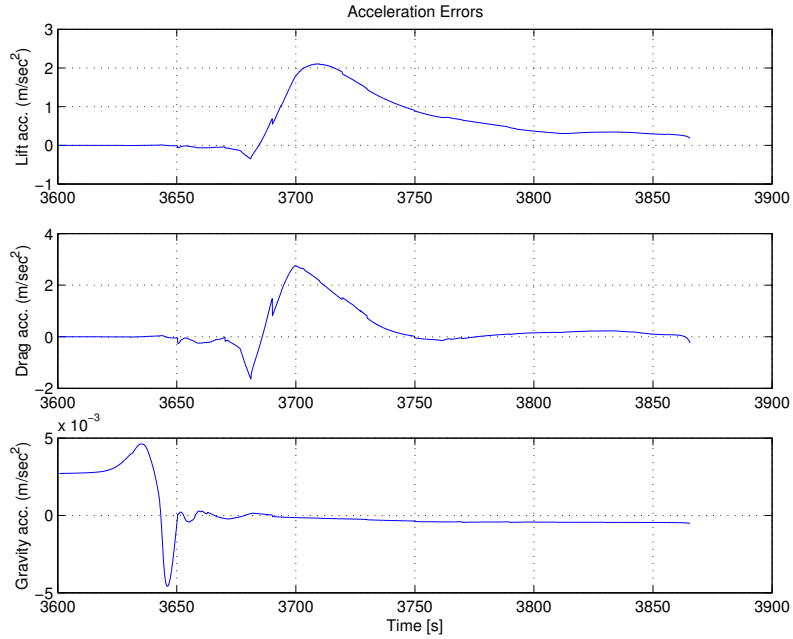

Figure 5.15: First filter bank: optimal wind frame acceleration estimation

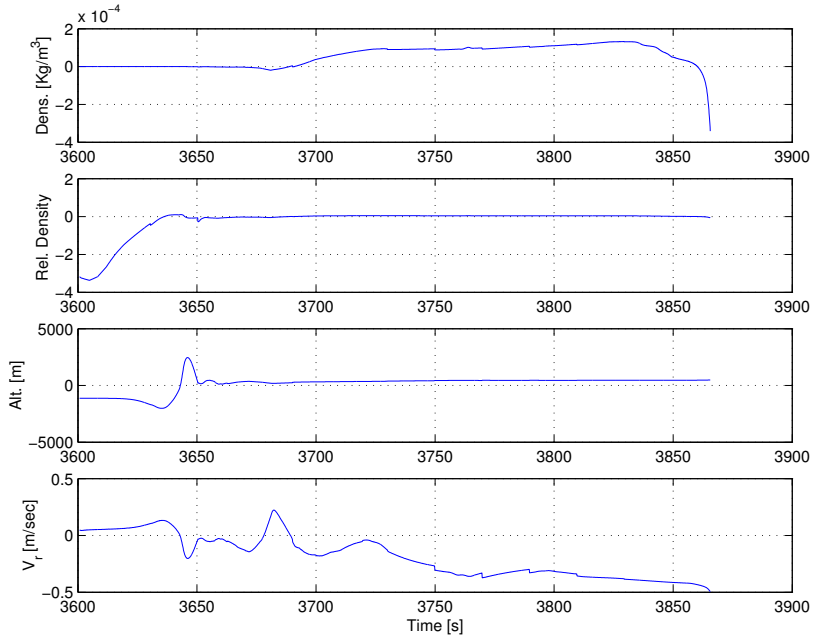

Figure 5.16: First filter bank: additional data

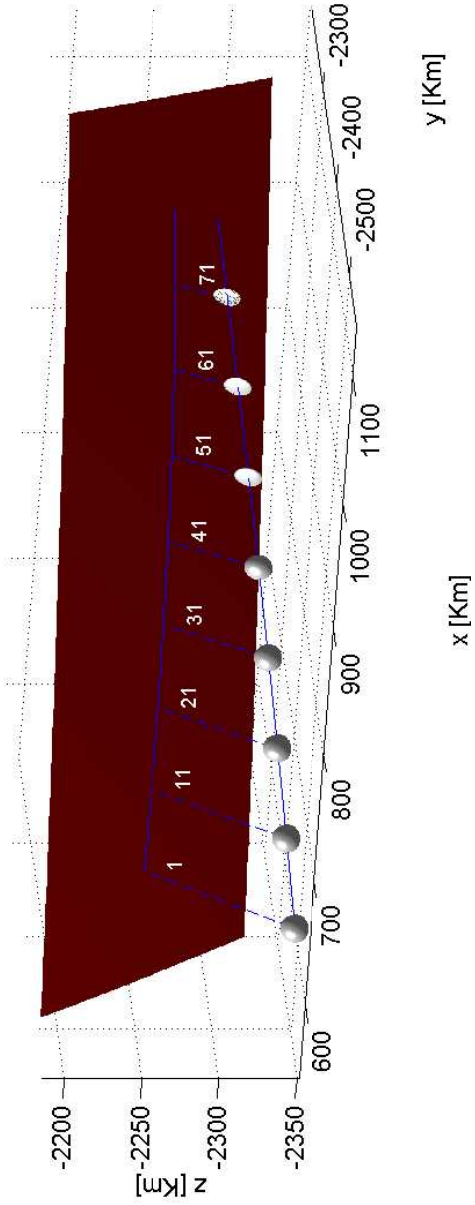

Figure 5.17: Position estimation error covariance evolution from beginning of simulation to 70*sec* (magnified 3 times). True trajectory and its projection on the surface of Mars is in blue. Seconds are labeled in white.

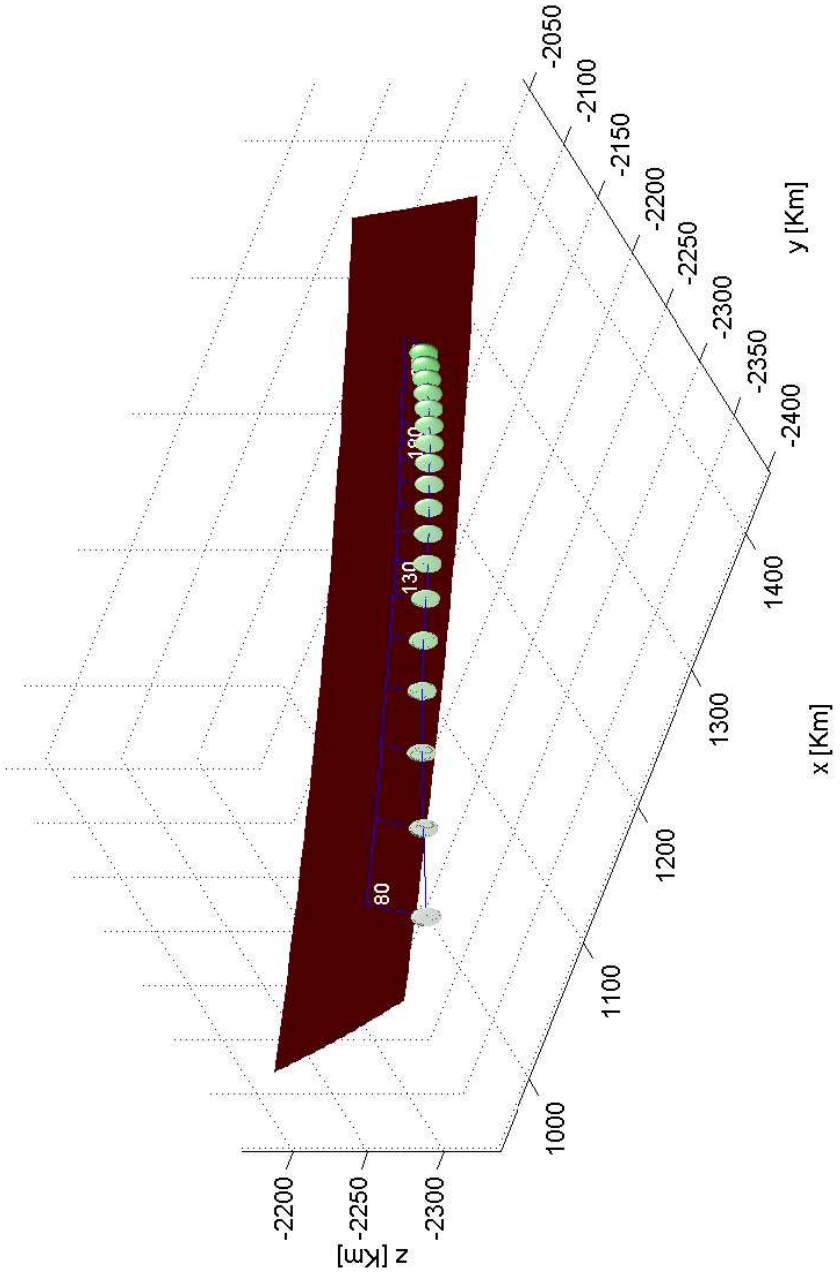

Figure 5.18: Position estimation error covariance evolution from 80*sec* to end of simulation (magnified 3 times). True trajectory and its projection on the surface of Mars is in blue. Seconds are labeled in white. The shade of green is proportional to the atmospheric density.

5.3.2 Second Filter Bank

In the previous example, one of the filter models (number 5) closely match a substantial part of the real density profile. Although this is the expected goal of the filter bank to achieve such a match by having a sufficient number of suitably chosen experts, it is necessary to check the performance of the mixture-of-experts when no such match exists, either through an insufficient number of experts, insufficient knowledge of the local atmospheric conditions, or unusual atmospheric conditions. To represent this, filter 5 was modified as follows: the reference density value ρ_i is now modified by a factor of 3.5 rather than 0.7, and the altitude scale factor is modified by 1.2. The resulting new density profile is somewhat closer and running parallel to the true profile at high altitude (see Fig. 5.19- 5.21) but diverges significantly at lower altitudes (Fig. 5.20- 5.22). This time, as expected, the final estimation errors are significantly worse : 5.659 Km in position, and 4.71 $m/sec.$ The gating network, however, correctly select filter 2 as the best performing, as shown in Fig. 5.23. We observe again a sudden degradation of the estimation performance of the mixture-of-experts at the very end of the navigation run as the density model errors increase.

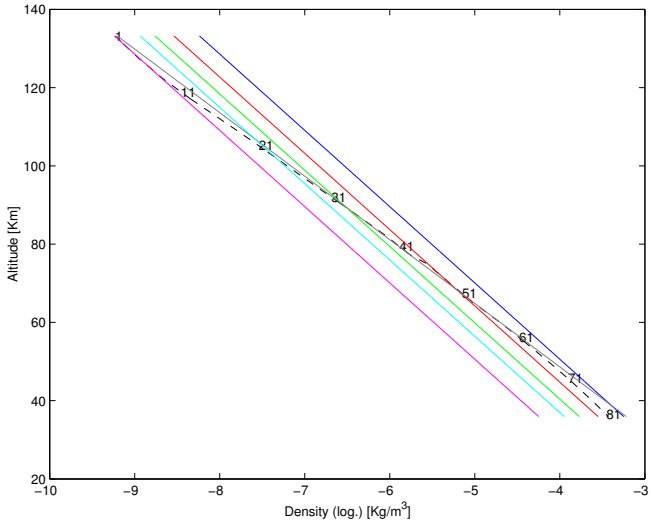

Figure 5.19: Second filter bank: actual and filter atmospheric density profiles down to 36 Km altitude (figures indicate time elapsed in seconds)

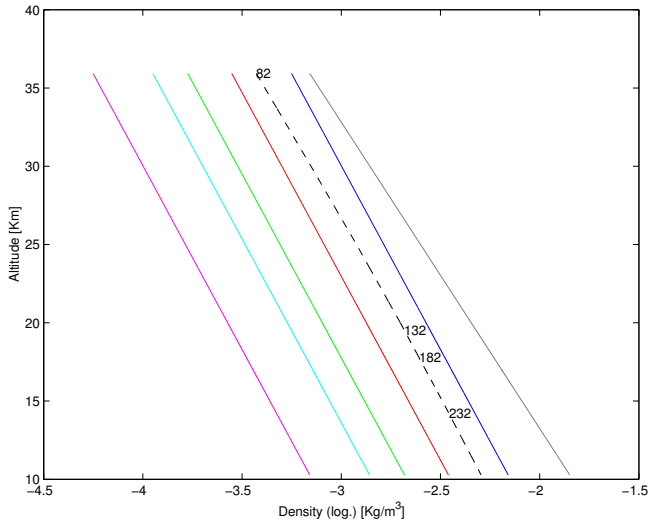

Figure 5.20: Second filter bank: actual and filter atmospheric density profiles down from 36 Km altitude down to parachute deployment

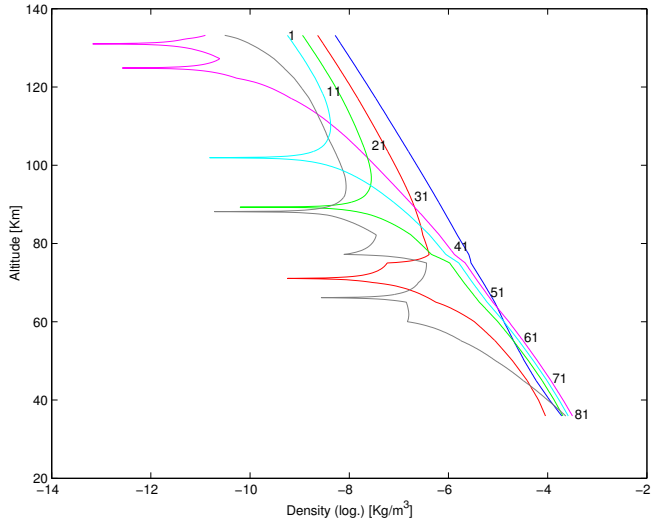

Figure 5.21: Second filter bank: differences between actual and filter atmospheric density profiles down to 36 Km altitude

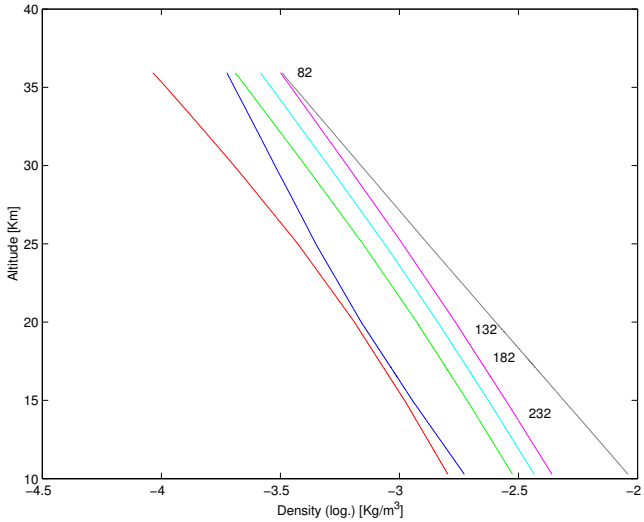

Figure 5.22: Second filter bank: differences between actual and filter atmospheric density profiles down from 36 Km altitude down to parachute deployment

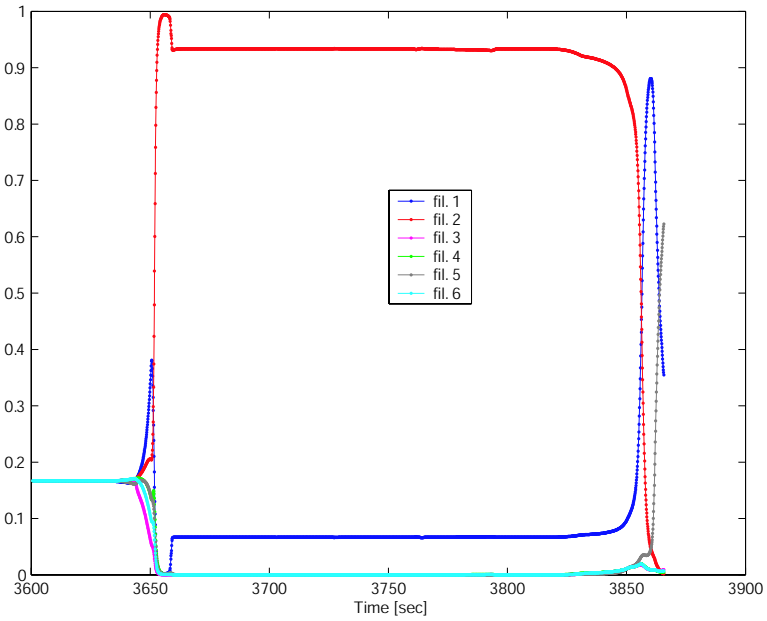

Figure 5.23: Second filter bank: gating network weights

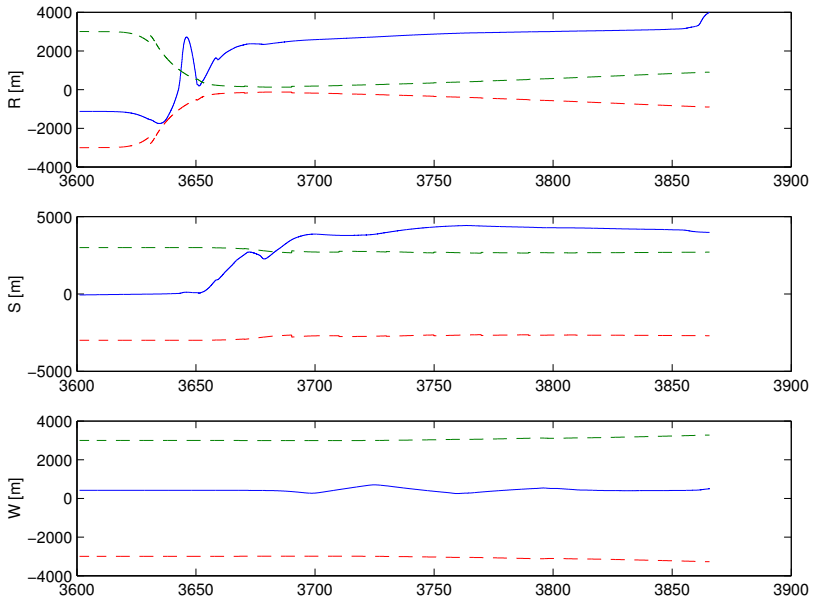

Figure 5.24: Second filter bank: optimal RSW position estimation, averaged over 30 Monte Carlo runs

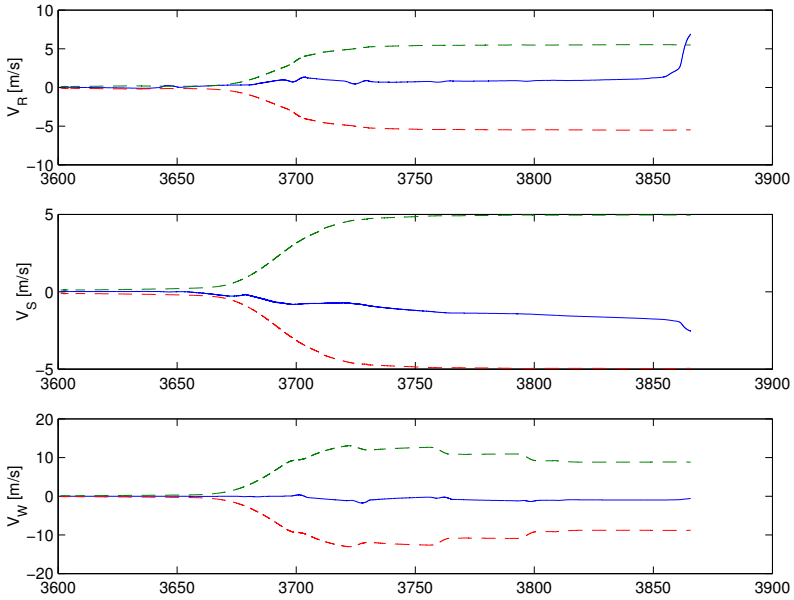

Figure 5.25: Second filter bank: optimal RSW velocity estimation

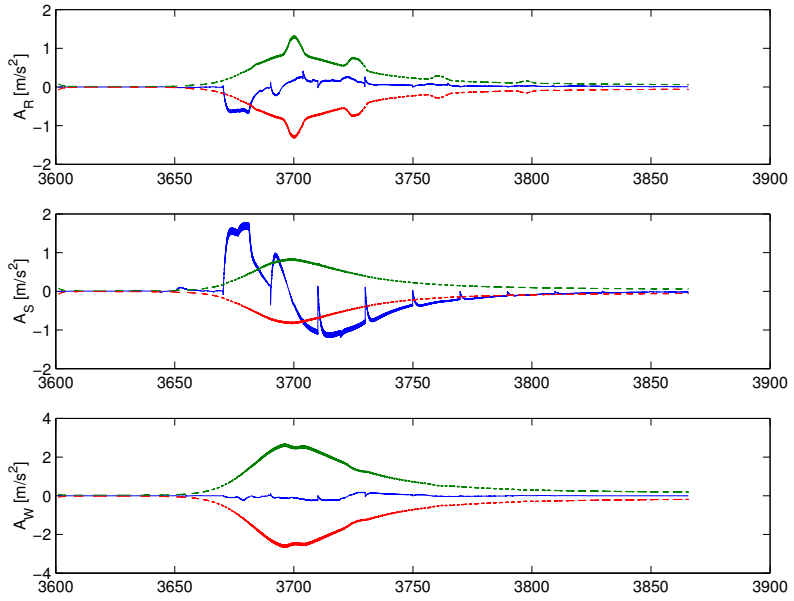

Figure 5.26: Second filter bank: optimal RSW acceleration estimation

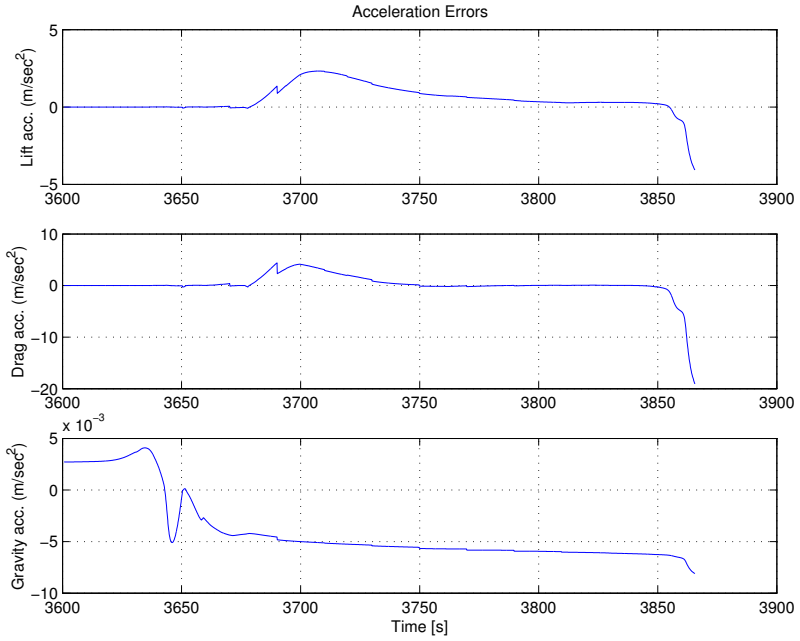

Figure 5.27: Second filter bank: optimal wind frame acceleration estimation

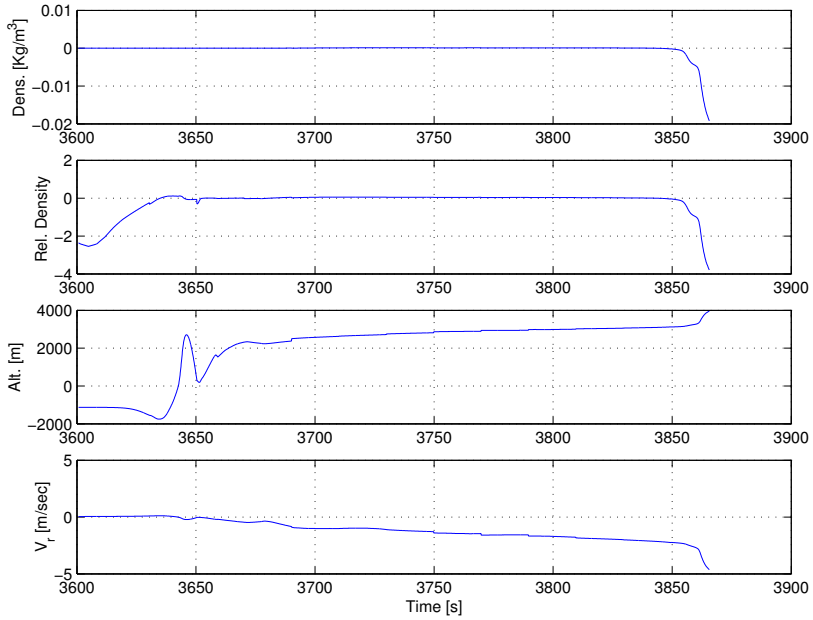

Figure 5.28: Second filter bank: additional data

5.3.3 Test of Robustness with Temporary Loss of IMU Data

The third and final test is a demonstration of the robustness of the model-based approach of the mixture-of-experts as opposed to the dead-reckoning scheme. A loss of IMU data, even temporary, can have disastrous consequences with the dead-reckoning method, while the mixture-of-experts can eventually recover. A complete gap in the stream of IMU data is the most likely incident, but since gyroscopic data are treated separately in this simulation, the navigation system cannot possibly recover from loss of angular rate information, thus here we will assume the IMU failure occurs only with the accelerometer data. The gap in these data is between $150sec$ and $165sec$ after the beginning of the simulation, just after the peak in acceleration.

The final position estimation error obtained with the mixture-of-experts is 5.397 Km versus 8.024 Km for dead-reckoning, and the final velocity estimation error is $2.518m/sec$ and $-46.604m/sec$ respectively. The estimation error in position and velocity obtained with dead-reckoning are shown in purple as comparison. It should be noted that the dead-reckoning results are also averaged over the same Monte Carlo runs, and that since dead-reckoning mostly propagates the original state estimation error without additional uncertainty other than the measured acceleration errors, the averaged state estimation error should be small. As we will see, this is not the case if there is a gap in the sequence of acceleration measurement, since there is no update process in the dead-reckoning scheme to recover from such events.

In Fig. 5.30 and 5.31, we can see that in the vertical channel ($\boldsymbol{R}$ direction) the dead-reckoning error remains constant, and does not seem to increase after IMU failure. This is because gravity acceleration is more important than aerodynamic acceleration in the vertical direction. In the $\boldsymbol{S}$ and $\boldsymbol{W}$ channels,

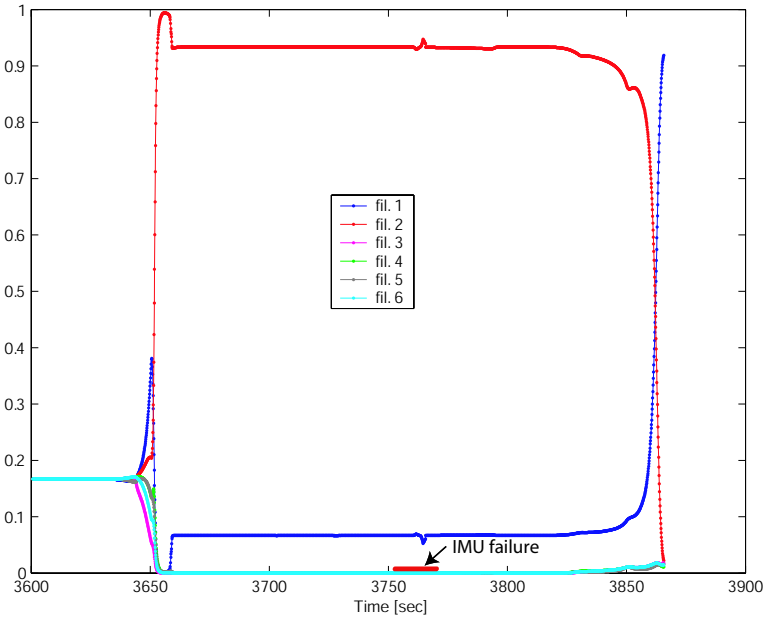

Figure 5.29: Temporary loss of data: gating network weights

however, the IMU failure create a permanent bias in the velocity estimate, and a corresponding ramp in the position estimate. The mixture-of-experts, however, performs in a similar way to the previous case, as can be seen from Fig. 5.24 and 5.25. The reaction and recovery of the filter bank to the IMU failure can be best seen in the acceleration estimate in Fig. 5.32 and the relative velocity estimate in Fig. 5.34.

5.4 Conclusions

The gating network regulated mixture-of-experts was tested in a simulation using data from a high-precision trajectory generation software. It was shown to be capable of correctly identifying the filter with the closest fitting atmospheric model, and by adequately judging all the filters it estimated the

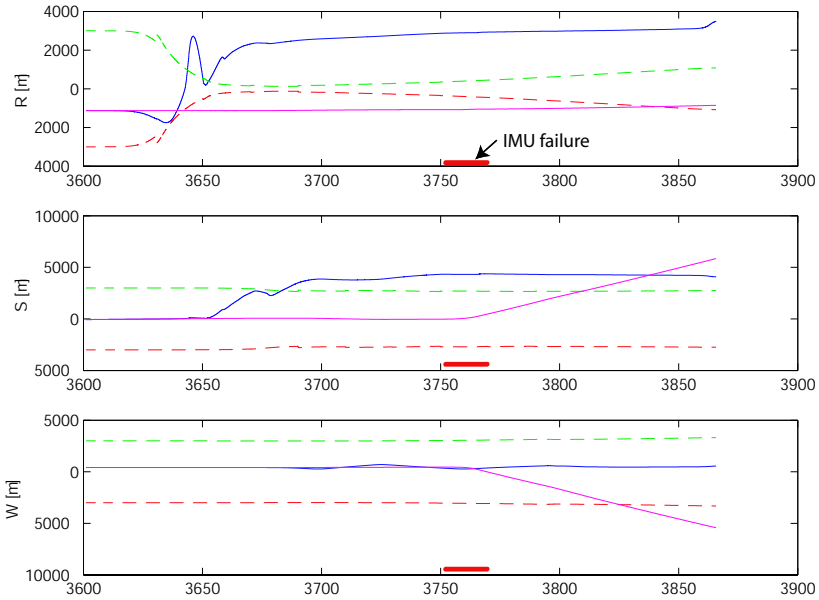

Figure 5.30: Temporary loss of data: optimal RSW position estimation, averaged over 30 Monte Carlo runs

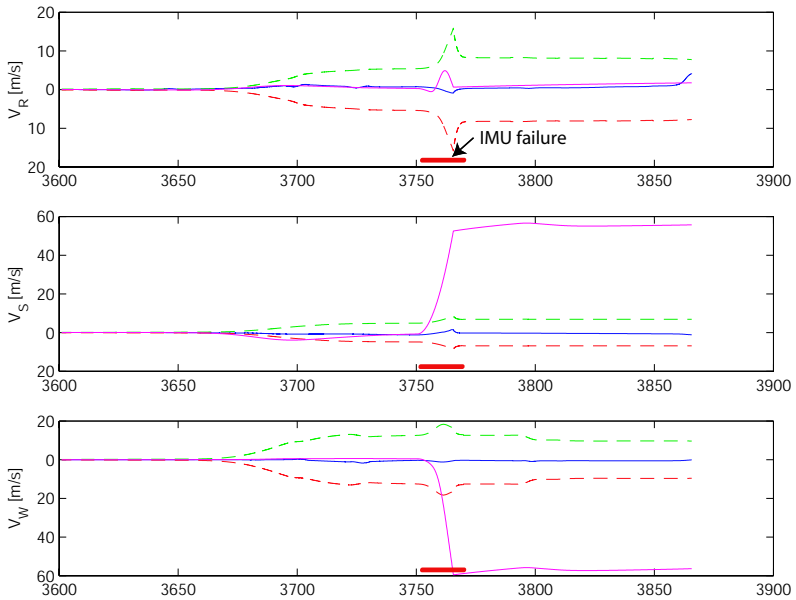

Figure 5.31: Temporary loss of data: optimal RSW velocity estimation

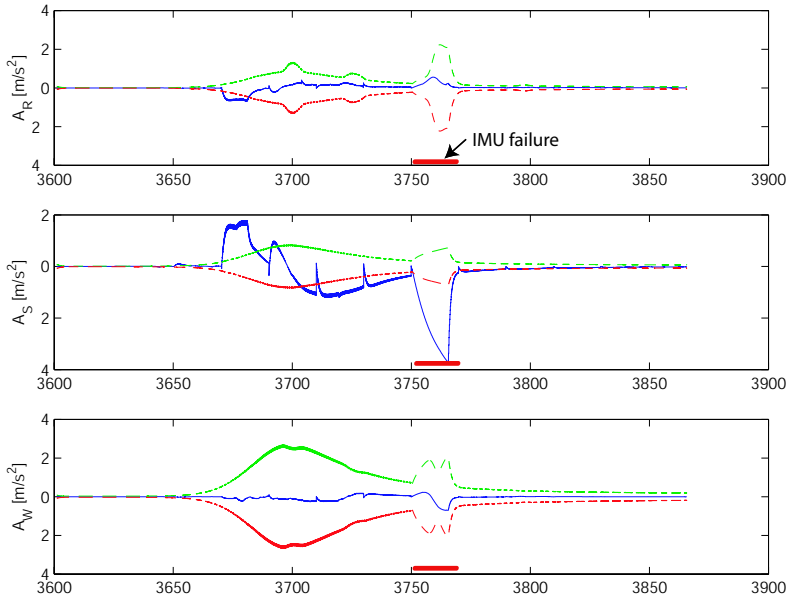

Figure 5.32: Temporary loss of data: optimal RSW acceleration estimation

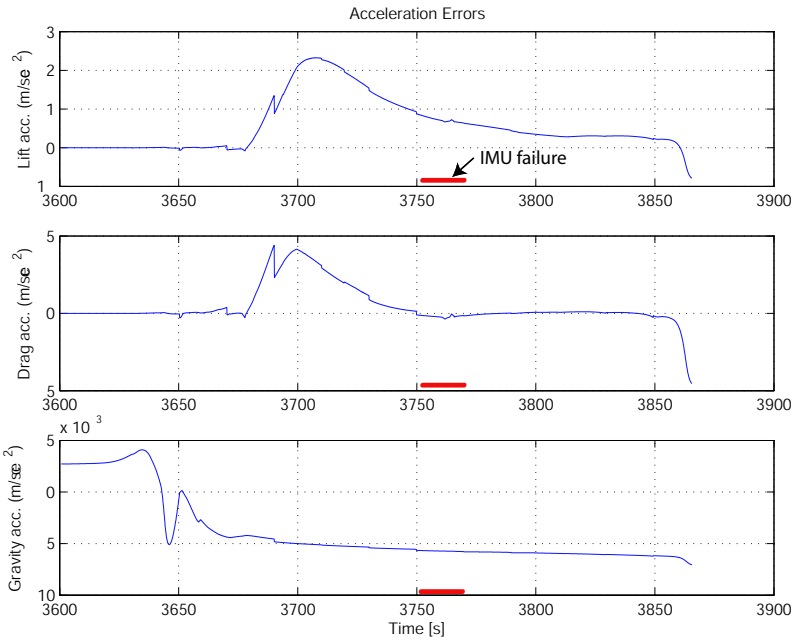

Figure 5.33: Temporary loss of data: optimal wind frame acceleration estimation

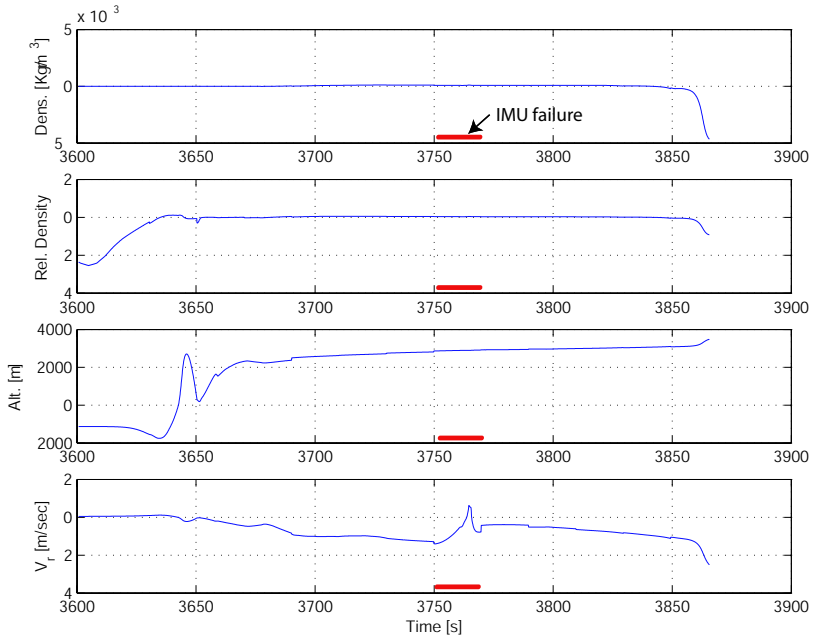

Figure 5.34: Temporary loss of data: additional data

position of the spacecraft to an accuracy between 1.6 Km and 5 Km, depending on whether or not one of the expert provided a close match to the density profile. It must also be noted that better results can be obtained if closer match are available for the last few seconds of navigation before parachute deployment. The mixture-of-experts also proved more robust to temporary loss of accelerometer data than the dead-reckoning method. It should be noted, however, that the mixture-of-experts seems to be operating at the threshold of unobservability.

These preliminary results should be completed by further studies and simulations, especially through additional layer of realism. The resolution of the spacecraft attitude should not be separated from the state estimation process. Gyroscopic data should be added to the measurement vector. Both

gyroscope and accelerometer bias and misalignments should be considered or even estimated. Observability of all these additional states however might imply the addition of new measurements. Dynamic pressure sensors such as the one used on Viking, and any type of future local Mars radio navigational helps are the most likely candidates.

A more complex atmosphere should also be taken into account. For example, unexpected high altitude winds should be modeled by some of the experts . This will require a more complex architecture of the mixture-of-experts, involving two or even more levels of regulation, such as the system described in [15] and [18].

In conclusion, the following improvements can be expected to make the regulated mixture-of-experts precision navigation system even more relevant for future planetary missions :

- augmentation of on-board computing power, allowing for larger filter banks and more complex models,
- availability of new sensors and measurement sources (atmospheric sensors, ground or on-orbit navigation aids), which can be readily incorporated into the estimation process
- further empirical knowledge of the structure and behavior of the visited planet atmosphere obtained from future missions, leading to more accurate atmospheric models.

Chapter 6

Conclusions

Our purpose here was to explore the possibilities offered in navigation techniques by multi-filter fusion techniques employing gating networks to compensate for uncertainties in the dynamic and environment models of single filters. In the two examples that were treated, a gating network was derived from the theory of neural networks to serve as a mediator to regulate a bank, or single layer of expert Extended Kalman Filters. The point of view and goal for each application was different (ground-based tracking and parameter identification in one, on-board precision navigation in the other) but the logic behind each was the same. In addition, in both cases the speed at which the gating network reached a decision (selection of a specific expert) or its capacity to make that decision without direct human intervention was important. Finally, both applications tested the effectiveness of the same dynamic flight model to two separate, but related problems.

In the tracking and identification of a maneuvering target from the ground, it was shown that identification of the target from a decoy was possible early in the trajectory, at high altitude where the aerodynamic effects are still small, using precision radar tracking, provided the objects have significants different ballistic coefficients. At lower altitude identified targets can be tracked using the same flight model used in the filter bank for identification. Investigation of this flight model also reveals certain behaviors in the torsion

of the trajectory which might be helpful for tracking and interception of the target.

The Mars entry navigation simulation demonstrated the capability of the mixture-of-experts to correctly identify the performance of the experts and to combine them to provide an estimate of the spacecraft states superior to what could be achieved by independent filters. The mixture-of-experts architecture was more robust than the usual dead-reckoning method, especially after considering IMU data gaps. Further improvements are needed, however, before this system can achieve the accuracy and the reliability necessary for precision Mars entry navigation. Among these are the increase in size of the filter bank, the refinement of the atmospheric density model, the inclusion of gyroscopic data as measurements and eventually the addition of external measurement sources.

Both examples seems to point toward the feasibility of applying gating network regulation to the highly dynamic and rapidly changing conditions of atmospheric entry, although further testing with increasingly realistic conditions is required before implementation of this technique. The main opportunity for improvement that should be the subject of further studies is the individual adaptability of each filter. In the previous work, adaptability was a feature of the system as a whole, where change was made through weight assignments to a discrete set of expert filters. But better performance is possible if, for instance, the best filter can be automatically tuned to better match the real dynamics and environments. Methods such as least squares and genetic algorithms have been employed, but mostly off-line, and not with such small reaction time (in the order of a few seconds) allowed. Such methods should be investigated and will become more feasible with advances in computing power,

especially for on-board applications.

Appendices

Appendix A

Derivation of the State Propagation Partials

The following is the derivation of the state propagation partials in the case where the state vector contains the inertial position $\boldsymbol{r}$, velocity $\boldsymbol{v}$ and aerodynamic acceleration $\boldsymbol{a}$. Derivation of the partials of the gravity vector $\boldsymbol{g}$ and the atmospheric density ρ are application dependent and will not be treated here. The subscript r designates quantities relative to the atmosphere of the planet, which is assumed to be immobile with respect to the surface of the planet.

First, let recall that the planetary rotation vector is

$$\boldsymbol{\Omega}_M = [0 \quad 0 \quad \Omega_M]^T.$$

This lead to the following definition for the relative state vector $\boldsymbol{X}_r$:

$$\begin{aligned} \boldsymbol{v}_r &= \boldsymbol{v} - \boldsymbol{\Omega}_M * \boldsymbol{r} \\ \boldsymbol{a}_r &= \boldsymbol{a} + \boldsymbol{g}(\boldsymbol{r}) - \boldsymbol{\Omega}_M * \boldsymbol{v} \\ \boldsymbol{X}_r &= [\boldsymbol{v}_r \quad \boldsymbol{a}_r]^T \end{aligned} \tag{A.1}$$

with

$$V_r = \|\boldsymbol{v}_r\| .$$

The wind frame is defined as follow:

$$\boldsymbol{e}_1^w = \frac{\boldsymbol{v}_r}{V_r} \qquad \boldsymbol{e}_2^w = -\frac{\frac{\boldsymbol{v}_r}{V_r} * \boldsymbol{r}}{\left\| \frac{\boldsymbol{v}_r}{V_r} * \boldsymbol{r} \right\|} \qquad \boldsymbol{e}_3^w = \boldsymbol{e}_1^w * \boldsymbol{e}_2^w . \tag{A.2}$$

We have then the equations of motion

$$\begin{aligned}
\dot{\boldsymbol{r}} &= \boldsymbol{v} \\
\dot{\boldsymbol{v}} &= \boldsymbol{a} + \boldsymbol{g} \\
\dot{\boldsymbol{a}} &= [\boldsymbol{\omega} * \boldsymbol{a}] + [\dot{\varphi}\ \boldsymbol{e}_1 * \boldsymbol{a}] - \left[\dot{D}\boldsymbol{e}_1\right] + \dot{L}\left[-\boldsymbol{e}_2 \sin\varphi + \boldsymbol{e}_3 \cos\varphi\right],
\end{aligned} \tag{A.3}$$

with

$$\boldsymbol{\omega} = \frac{\boldsymbol{v}_r \times \boldsymbol{a}_r}{\|\boldsymbol{v}_r\|^2}. \tag{A.4}$$

The first step is the derivation of the partials for the relative states $\boldsymbol{X}_r$:

$$\begin{aligned}
\frac{\partial \boldsymbol{X}_r}{\partial \boldsymbol{X}} &= \begin{bmatrix} \boldsymbol{S}(\boldsymbol{\Omega}_M) & \boldsymbol{I}_{3\times3} & \boldsymbol{0}_{3\times3} \\ \boldsymbol{0}_{3\times3} & \boldsymbol{S}(\boldsymbol{\Omega}_M) & \boldsymbol{I}_{3\times3} \end{bmatrix} \\
\frac{\mathrm{d}\boldsymbol{X}_r}{\mathrm{d}\boldsymbol{g}} &= [\boldsymbol{0}_{3\times3} \quad \boldsymbol{I}_{3\times3}] \\
\frac{\mathrm{d}\boldsymbol{X}_r}{\mathrm{d}\boldsymbol{X}} &= \frac{\partial \boldsymbol{X}_r}{\partial \boldsymbol{X}} + \frac{\mathrm{d}\boldsymbol{X}_r}{\mathrm{d}\boldsymbol{g}} \frac{\mathrm{d}\boldsymbol{g}}{\mathrm{d}\boldsymbol{X}},
\end{aligned} \tag{A.5}$$

where $\boldsymbol{S}()$ designates the skew symmetric representing the vector cross product in the sense that

$$\boldsymbol{a} * \boldsymbol{b} = \boldsymbol{S}(\boldsymbol{a})\boldsymbol{b} \quad \forall \quad \boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^3.$$

Furthermore, we can now compute the partials of each of the wind frame vectors (all superscripts w have been dropped for easier reading):

$$
\begin{aligned}
\frac{\mathrm{d}\boldsymbol{e}_1}{\mathrm{d}\boldsymbol{X}_r} &= \frac{1}{V_r^3}\begin{bmatrix} V_r^2 - v_{r_x}^2, -v_{r_x}v_{r_y}, & -v_{r_x}v_{r_z} & \\ -v_{r_x}v_{r_y}, V_r^2 - v_{r_y}^2, & -v_{r_y}v_{r_z} & \mathbf{0}_{3\times 3} \\ -v_{r_x}v_{r_z}, -v_{r_y}v_{r_z}, & V_r^2 - v_{r_z}^2 & \end{bmatrix} \\
\frac{\partial \boldsymbol{e}_2}{\partial \boldsymbol{e}_1} &= -\left(-\frac{\boldsymbol{S}(\boldsymbol{r})}{\|\boldsymbol{e}_1 * \boldsymbol{r}\|} + 2\frac{(\boldsymbol{e}_1 * \boldsymbol{r})(\boldsymbol{S}(\boldsymbol{r})(\boldsymbol{e}_1 * \boldsymbol{r}))^T}{\|\boldsymbol{e}_1 * \boldsymbol{r}\|^3}\right) \\
\frac{\partial \boldsymbol{e}_3}{\partial \boldsymbol{e}_1} &= \boldsymbol{S}(\boldsymbol{e}_2) \\
\frac{\partial \boldsymbol{e}_2}{\partial \boldsymbol{X}} &= -\left[-\frac{\boldsymbol{S}(\boldsymbol{e}_1)}{\|\boldsymbol{e}_1 * \boldsymbol{r}\|} + 2\frac{(\boldsymbol{e}_1 * \boldsymbol{r})(\boldsymbol{S}(\boldsymbol{e}_1)(\boldsymbol{e}_1 * \boldsymbol{r}))^T}{\|\boldsymbol{e}_1 * \boldsymbol{r}\|^3} \quad \mathbf{0}_{3\times 6}\right] \\
\frac{\mathrm{d}\boldsymbol{e}_1}{\mathrm{d}\boldsymbol{X}} &= \frac{\partial \boldsymbol{e}_1}{\partial \boldsymbol{X}_r}\frac{\partial \boldsymbol{X}_r}{\partial \boldsymbol{X}} \\
\frac{\mathrm{d}\boldsymbol{e}_2}{\mathrm{d}\boldsymbol{X}} &= \frac{\partial \boldsymbol{e}_2}{\partial \boldsymbol{X}} + \frac{\partial \boldsymbol{e}_2}{\partial \boldsymbol{e}_1}\frac{\mathrm{d}\boldsymbol{e}_1}{\mathrm{d}\boldsymbol{X}} \\
\frac{\mathrm{d}\boldsymbol{e}_3}{\mathrm{d}\boldsymbol{X}} &= \frac{\partial \boldsymbol{e}_3}{\partial \boldsymbol{e}_1}\frac{\partial \boldsymbol{e}_1}{\partial \boldsymbol{X}}.
\end{aligned}
\tag{A.6}
$$

Partials are now taken for each of the term in the equation of motion A.3:

$$
\begin{aligned}
\boldsymbol{T}_a &= \boldsymbol{\omega} * \boldsymbol{a} \\
\frac{\partial \boldsymbol{T}_a}{\partial \boldsymbol{X}} &= [\mathbf{0}_{3\times 6} \quad \boldsymbol{S}(\omega)] \\
\frac{\mathrm{d}\boldsymbol{\omega}}{\mathrm{d}\boldsymbol{X}_r} &= \frac{1}{V_r^2}[-\boldsymbol{S}(\boldsymbol{a}_r) \quad \boldsymbol{S}(\boldsymbol{v}_r)] - 2\frac{\boldsymbol{v}_r * \boldsymbol{a}_r}{V_r^4}\begin{bmatrix}\boldsymbol{v}_r^T & \mathbf{0}_{1\times 3}\end{bmatrix} \\
\frac{\partial \boldsymbol{T}_a}{\partial \boldsymbol{\omega}} &= -\boldsymbol{S}(\boldsymbol{a}) \\
\frac{\mathrm{d}\boldsymbol{T}_a}{\mathrm{d}\boldsymbol{X}} &= \frac{\partial \boldsymbol{T}_a}{\partial \boldsymbol{X}} + \frac{\partial \boldsymbol{T}_a}{\partial \boldsymbol{\omega}}\frac{\mathrm{d}\boldsymbol{\omega}}{\mathrm{d}\boldsymbol{X}_r}\frac{\mathrm{d}\boldsymbol{X}_r}{\mathrm{d}\boldsymbol{X}},
\end{aligned}
\tag{A.7}
$$

$$
\begin{aligned}
\boldsymbol{T}_b &= \dot{\phi}\boldsymbol{e}_1 * \boldsymbol{a} \\
\frac{\partial \boldsymbol{T}_b}{\partial \boldsymbol{X}} &= \dot{\phi}\left[\boldsymbol{0}_{3\times 6} \quad \boldsymbol{S}(\boldsymbol{e}_1)\right] \\
\frac{\partial \boldsymbol{T}_b}{\partial \boldsymbol{e}_1} &= -\dot{\phi}\boldsymbol{S}(\boldsymbol{a}) \\
\frac{\mathrm{d}\boldsymbol{T}_b}{\mathrm{d}\boldsymbol{X}} &= \frac{\partial \boldsymbol{T}_b}{\partial \boldsymbol{X}} + \frac{\partial \boldsymbol{T}_b}{\partial \boldsymbol{e}_1}\frac{\mathrm{d}\boldsymbol{e}_1}{\mathrm{d}\boldsymbol{X}_r}\frac{\mathrm{d}\boldsymbol{X}_r}{\mathrm{d}\boldsymbol{X}},
\end{aligned}
\tag{A.8}
$$

$$
\begin{aligned}
\boldsymbol{T}_c &= -\dot{D}\boldsymbol{e}_1 \\
\frac{\mathrm{d}\boldsymbol{T}_c}{\mathrm{d}\boldsymbol{X}} &= -\boldsymbol{e}_1\frac{\mathrm{d}\dot{D}}{\mathrm{d}\boldsymbol{X}} - \dot{D}\frac{\mathrm{d}\boldsymbol{e}_1}{\mathrm{d}\boldsymbol{X}},
\end{aligned}
\tag{A.9}
$$

$$
\begin{aligned}
\boldsymbol{T}_d &= \dot{L}\left(-\boldsymbol{e}_2\sin(\phi) + \boldsymbol{e}_3\cos(\phi)\right) \\
\frac{\mathrm{d}\boldsymbol{T}_d}{\mathrm{d}\boldsymbol{X}} &= \left(-\boldsymbol{e}_2\sin(\phi) + \boldsymbol{e}_3\cos(\phi)\right)\frac{\mathrm{d}\dot{L}}{\mathrm{d}\boldsymbol{X}} + \\
&\quad \dot{L}\left(-\boldsymbol{e}_2\sin(\phi) + \boldsymbol{e}_3\cos(\phi)\right),
\end{aligned}
\tag{A.10}
$$

Detailed expressions of $\dot{L}$ and $\dot{D}$ are given by

$$\dot{L} = \frac{S}{m}\left(\dot{C}_L q + C_L\dot{q}\right) \tag{A.11}$$

$$\dot{D} = \frac{S}{m}\left(\dot{C}_D q + C_D\dot{q}\right) \tag{A.12}$$

with

$$q = \frac{1}{2}\rho V_r^2 \tag{A.13}$$

$$\dot{q} = \frac{1}{2}\dot{\rho}V_r^2 + \rho\boldsymbol{a}_r \odot \boldsymbol{v}_r, \tag{A.14}$$

Hence, we have the following partials:

$$
\begin{aligned}
\frac{\partial q}{\partial \boldsymbol{X}} &= \frac{1}{2}\frac{\mathrm{d}q}{\mathrm{d}\boldsymbol{X}}\rho V_r^2 \\
\frac{\mathrm{d}q}{\mathrm{d}\boldsymbol{X}_r} &= \rho\left[\boldsymbol{v}_r^T \quad \boldsymbol{0}_{1\times 3}\right] \\
\frac{\mathrm{d}q}{\mathrm{d}\boldsymbol{X}} &= \frac{\partial q}{\partial \boldsymbol{X}} + \frac{\mathrm{d}q}{\mathrm{d}\boldsymbol{X}_r}\frac{\mathrm{d}\boldsymbol{X}_r}{\mathrm{d}\boldsymbol{X}},
\end{aligned}
\tag{A.15}
$$

$$
\begin{aligned}
\frac{\partial \dot{q}}{\partial \boldsymbol{X}} &= \frac{1}{2}\frac{\mathrm{d}\dot{\rho}}{\mathrm{d}\boldsymbol{X}}V_r^2 + \frac{\mathrm{d}\rho}{\mathrm{d}\boldsymbol{X}}\boldsymbol{a}_r \odot \boldsymbol{v}_r \\
\frac{\mathrm{d}\dot{q}}{\mathrm{d}\boldsymbol{X}_r} &= \dot{\rho}\begin{bmatrix}\boldsymbol{v}_r^T & \boldsymbol{0}_{1\times 3}\end{bmatrix} + \rho\begin{bmatrix}\boldsymbol{a}_r^T & \boldsymbol{v}_r^T\end{bmatrix} \\
\frac{\mathrm{d}\dot{q}}{\mathrm{d}\boldsymbol{X}} &= \frac{\partial \dot{q}}{\partial \boldsymbol{X}} + \frac{\mathrm{d}\dot{q}}{\mathrm{d}\boldsymbol{X}_r}\frac{\mathrm{d}\boldsymbol{X}_r}{\mathrm{d}\boldsymbol{X}},
\end{aligned}
\tag{A.16}
$$

$$
\begin{aligned}
\frac{\mathrm{d}\dot{L}}{\mathrm{d}\boldsymbol{X}} &= \frac{S}{m}\left(C_L\frac{\mathrm{d}\dot{q}}{\mathrm{d}\boldsymbol{X}} + \dot{C}_L\frac{\mathrm{d}q}{\mathrm{d}\boldsymbol{X}}\right) \\
\frac{\mathrm{d}\dot{D}}{\mathrm{d}\boldsymbol{X}} &= \frac{S}{m}\left(C_D\frac{\mathrm{d}\dot{q}}{\mathrm{d}\boldsymbol{X}} + \dot{C}_D\frac{\mathrm{d}q}{\mathrm{d}\boldsymbol{X}}\right).
\end{aligned}
\tag{A.17}
$$

Finally, the state propagation partial matrix $\boldsymbol{F}$ is given by

$$
\boldsymbol{F} = \begin{bmatrix}
\boldsymbol{0}_{3\times 3} & \boldsymbol{I}_{3\times 3} & \boldsymbol{0}_{3\times 3} \\
\boldsymbol{0}_{3\times 3} & \boldsymbol{0}_{3\times 3} & \boldsymbol{I}_{3\times 3} \\
 & \frac{\mathrm{d}\boldsymbol{T}_a}{\mathrm{d}\boldsymbol{X}} + \frac{\mathrm{d}\boldsymbol{T}_b}{\mathrm{d}\boldsymbol{X}} + \frac{\mathrm{d}\boldsymbol{T}_c}{\mathrm{d}\boldsymbol{X}} + \frac{\mathrm{d}\boldsymbol{T}_b}{\mathrm{d}\boldsymbol{X}} &
\end{bmatrix}.
\tag{A.18}
$$

Bibliography

[1] Tauber, M. E., Bowles, J. V., and Yang, L., "Atmospheric Environment During Maneuvering Descent from Martian Orbit," *Journal of Spacecraft and Rockets*, Vol. 26, No. 5, September-October 1989, pp. 330–337.

[2] Kalman, R. E. and Bucy, R. S., "New Results in Linear Filtering and Prediction Theory," *Transactions of the ASME, Serie D, Journal of Basic Engineering*, March 1961, pp. 95–108.

[3] Kalman, R. E., "New Methods and Results in Linear Prediction and Filtering Theory," *Proc. Symp. on Engineering Applications of Random Function Theory and Probability*, edited by Wiley, 1961.

[4] Price, C. F., "An Analysis of the Divergence Problem in the Kalman Filter," *IEEE Transactions on Automatic Control*, Vol. 13, December 1968, pp. 699–702.

[5] Fitzgerald, R. J., "Divergence of Kalman Filter," *IEEE Transactions on Automatic Control*, Vol. 16, December 1971, pp. 736–747.

[6] Regan, F. J. and Anandakrishnan, S. M., *Dynamics of Atmospheric Re-Entry*, American Institute of Aeronautics and Astronautics, Washington, DC, 1993.

[7] Mehra, R. K., "Approaches to Adaptive Filtering," *IEEE Transactions on Automatic Control*, Vol. 17, October 1972, pp. 693–698.

[8] Magill, D. T., "Optimal Adaptive Estimation of Sampled Stochastic Processes," *IEEE Transactions on Automatic Control*, Vol. AC-10, No. 4, October 1965, pp. 434–439.

[9] Brown, R. G. and Hwang, P. Y. C., *Introduction to Random Signals and Applied Kalman Filtering*, John Wiley & Sons, Inc., New York, Chichester, Brisbane, Toronto, Singapore, 1992.

[10] Jacobs, R. and Jordan, M. I., "A Competitive Modular Connectionist Architecture," *Advances in Neural Information Processing Systems 3*, edited by R. P. Lippmann, J. E. Moody, and D. J. Touretzky, San Mateo, CA: Morgan Kaufmann, 1991, pp. 767–773.

[11] Jacobs, R., Jordan, M. I., Nowlan, S. J., and Hinton, G. E., "Adaptive Mixtures of Local Experts," *Neural Computation 3*, 1991, pp. 79–87.

[12] Jacobs, R., Jordan, M. I., and Barto, A. G., "Task Decomposition Through Competion in a Modular Connectionist Architecture: The What and Where Vision Tasks," *Cognitive Science 15*, 1991, pp. 219–250.

[13] Jordan, M. I. and Jacobs, R., "Hierarchies of Adaptive Experts," *Advances in Neural Information Processing Systems 4*, edited by R. P. Lippmann, J. E. Moody, and D. J. Touretzky, San Mateo, CA: Morgan Kaufmann, 1992, pp. 985–992.

[14] Jacobs, R. and Jordan, M. I., "Learning Piecewise Control Strategies in a Modular Neural Network Architecture," *IEEE Transactions on Systems, Man, and Cybernetics*, , No. 23, 1993, pp. 337–345.

[15] Chaer, W. S., *A Mixture-of-Experts Approach to Adaptive Estimation*, Ph.D. thesis, The University of Texas at Austin, 1996.

[16] Chaer, W. S., Bishop, R. H., and Ghosh, J., "A Mixture-of-Experts Framework for Adaptive Kalman Filtering," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 27, No. 3, June 1997, pp. 452–464.

[17] Chaer, W. S., Bishop, R. H., and Ghosh, J., "Hierarchical Adaptive Kalman Filtering for Interplanetary Orbit Determination," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 34, No. 3, July 1998, pp. 883–896.

[18] Crain, T. P., *Adaptive Interplanetary Orbit Determination*, Ph.D. thesis, The University of Texas at Austin, 2000.

[19] Lewis, G. N. and Postol, T. A., "Future Challenges to Ballistic Missile Defense," *IEEE Spectrum*, September 1997, pp. 60–68.

[20] Bishop, R. H., Dubois-Matra, O., and Ely, T., "Robust Entry Navigation Using Hierarchical Filter Architectures Regulated with Gating Networks," *16th International Symposium on Space Flight Dynamics*, 2001.

[21] Gelb, A., editor, *Applied Optimal Estimation*, Massachusetts Institute of Technoly Press, Cambridge, Massachusetts, and London, England, 1974.

[22] Haykin, S., *Neural Networks, A Comprehensive Foundation*, Macmillan College Publishing Company, New York, 1994.

[23] Widrow, B. and Lehr, M. A., "30 Years of Adaptive Neural Networks: Perceptron, Madaline and Backpropagation," *Proceedings of the IEEE*, Vol. 78, No. 9, September 1990, pp. 1415–1442.

[24] Bridle, J. S., "Probabilistic Interpretation of Feedforward Classification Network Outputs, with Relationships to Statistical Pattern Recognition,"

Neuro-computing: Algorithms Architectures and Applications, edited by F. Fougelman-Soulie and J. Hérault, Springer-Verlag, New York, 1990, pp. 227–236.

[25] Bridle, J. S., "Training Stochastic Model Recognition Algorithms as Networks can Lead to Maximum Mutual Information Estimation of Parameters," *Advances in Neural Information Processing Systems 2*, edited by D. J. Touretzky, Morgan Kaufmann, San Mateo, CA, 1990, pp. 211–217.

[26] Shynk, J. J., "Performance Surfaces of a Single-Layer Perceptron," *IEEE Transactions on Neural Networks*, Vol. 1, No. 3, September 1990, pp. 268–274.

[27] Bishop, R. H. and Antoulas, A. C., "Nonlinear Approach to Aircraft Tracking Problem," *Journal of Guidance, Control and Dynamics*, Vol. 17, No. 5, September-October 1994, pp. 1124–1130.

[28] Mehrotra, K. and Mahapatra, P. R., "A Jerk Model for Tracking Highly Maneuvering Targets," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 33, No. 4, October 1997, pp. 1094–1105.

[29] Kaula, W. M., *Theory of Satellite Geodesy*, Blaisdell Publishing Company, Waltham, Massachusetts, Toronto, London, 1966.

[30] Cardillo, G. P., Mrstik, A. V., and Plambeck, T., "A Track Filter for Reentry Objects with Uncertain Drag," *IEEE Transactions on Aerospace and Electronic Systems*, Vol. 35, No. 2, April 1999, pp. 394–408.

[31] Savage, P. G., "Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms," *Journal of Guidance, Control and Dynamics*, Vol. 21, No. 1, January-February 1998, pp. 19–28.

[32] Savage, P. G., "Strapdown Inertial Navigation Integration Algorithm Design Part 2: Velocity and Position Algorithms," *Journal of Guidance, Control and Dynamics*, Vol. 21, No. 2, March-April 1998, pp. 208–221.

[33] O'Neill, B., *Elementary Differential Geometry*, Harcourt Brace Jovanovich, 1966.

[34] Braun, R. D., Spencer, D. A., Kallemeyn, P. H., and Vaughan, R. M., "Mars Pathfinder Atmospheric Entry Navigation Operations," *Journal of Spacecraft and Rockets*, Vol. 36, No. 3, May-June 1999, pp. 348–356.

[35] "Guidance and Navigation for Entry Vehicles," SP 8015, NASA, 1968.

[36] Morth, R., "Apollo Guidance and Navigation: Reentry Guidance for Apollo," Technical Paper R-532, MIT Instrumentation Laboratory, Cambridge 39, Massachusetts, 1966.

[37] Blanchard, R. C. and Walberg, G. D., "Determination of the Hypersonic-Continuum / Rarefied-Flow Drag Coefficient of the Viking Lander Capsule 1 Aeroshell From Flight Data," Technical Paper 1793, NASA, 1980.

[38] Desai, P. N., Micheltree, R. A., and Cheatwood, F. M., "Entry Dispersion Analysis for the Stardust Comet Sample Return Capsule," *Journal of Spacecraft and Rockets*, Vol. 36, No. 3, May-June 1999, pp. 463–469.

[39] Justus, C. G. and Johnson, D. L., "Mars Global Reference Atmospheric Model 2001 Version (MARS-GRAM 2001): Users Guide," Technical Memorandum 210961, NASA, 2001.

[40] Blanchard, R. C., Wilmoth, R. G., and Moss, J. N., "Aerodynamic Flight Measurements and Rarefied-Flow Simulations of Mars Entry Vehicles," *Journal of Spacecraft and Rockets*, Vol. 34, No. 5, ... 1997, pp. 687–690.

[41] Vallado, D. A., *Fundamentals of Astrodynamics and Applications*, The McGraw-Hill Companies, Inc., 1997.

Vita

Olivier Dubois-Matra, the son of Bernard and Jeanine Dubois-Matra, was born on February 25, 1973, in Paris, France, but considers the village of Leugny, in Burgundy, his hometown. After getting his Baccalaureat (high school diploma) at the *Lycée* Jacques Amyot in Auxerre, France, in 1990 he attended Preparatory Classes at the same school. He was admitted in 1992 at the *Ecole Centrale de Lille*, from which he obtained his engineering degree in 1995. He joined the Graduate School at the University of Texas in the Aerospace Engineering Department through an exchange program in August 1994, and after obtaining his Master of Science degree there in 1996 has been pursuing a Ph.D in the area of Guidance, Navigation and Control.

Permanent address: 600 e. 53rd Street, Apt. 244
Austin, Texas 78751

This dissertation was typeset with LaTeX[†] by the author.

[†]LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's TeX Program.